ONLINE FOOD DELIVERY SYSTEM

A Case Study Submitted to

DEPARTMENT
of
COMPUTER SCIENCE AND SYSTEMS ENGINEERING

Submitted by

B.V.S.S.GANESH	**20121A2910**
M.V.S.PRANAY	**20121A2941**
G.SAI KUMAR	**20121A2917**
M.SATHWIK	**20121A2937**

Under the Guidance of
M.RAMU, M.Tech
Assistant Professor
Dept. of CSSE, SVEC

Department of Computer Science and Systems Engineering

Sree Vidyanikethan Engineering College (Autonomous)
Sree Sainath Nagar, Tirupati – 517 102
(2021-2022)

SREE VIDYANIKETHAN ENGINEERING COLLEGE

(AUTONOMOUS)
Sree Sainath Nagar, Tirupati

DEPARTMENT OF COMPUTER SCIENCE AND SYSTEMS ENGINEERING

CERTIFICATE

This is to certify that the case study report entitled

ONLINE FOOD DELIVERY SYSTEM

is the Bonafide work done by

B.V.S.S.GANESH	**20121A2910**
M.V.S.PRANAY	**20121A2941**
G.SAI KUMAR	**20121A2917**
M.SATHWIK	**20121A2937**

in the Department of **Computer Science and Systems Engineering**, and submitted to Computer Science and Systems Engineering during the academic year 2021-2022. This work has been carried out under my supervision.

Guide:	**Head:**
M.Ramu	Dr. K. Ramani
Assistant Professor	Professor & Head
Dept. of CSSE	Dept. of CSSE

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DEPARTMENT OF COMPUTER SCIENCE AND SYSTEMS ENGINEERING

VISION

To become a centre of excellence in Computer Sciences and Systems Engineering through teaching, training, research and innovation to create quality engineering professionals who can solve the growing complex problems of the society.

MISSION

- Established with the cause of development of technical education in advanced computer sciences and engineering with applications to systems there by serving the society and nation.

- Transfer of Knowledge through contemporary curriculum and fostering faculty and student development.

- Create keen interest for research and innovation among students and faculty by understanding the needs of the society and industry.

- Skill development among diversity of students in technical domains and profession for development of systems and processes to meet the demands of the industry and research

- Imbibing values and ethics in students for prospective and promising engineering profession and develop a sense of respect for all.

PROGRAM EDUCATIONAL OBJECTIVES

1. Demonstrate competencies in the Computer Science domain and Management with an ability to comprehend, analyze, design and create software systems for pursuing advanced studies in the areas of interest.

2. Evolve as entrepreneurs or be employed by acquiring required skill sets for developing computer systems and solutions in multi-disciplinary areas.

3. Exhibit progression and professional skill development in Computer programming and systems development with ethical attitude through life-long learning.

PROGRAM SPECIFIC OUTCOMES

PSO1: Employ Systems Approach to model the solutions for real life problems, design and develop software systems by applying Modern Tools.

PSO2: Develop solutions using novel algorithms in High Performance Computing and Data Science.

PSO3: Use emerging technologies for providing security and privacy to design, deploy and manage network systems.

.

PROGRAM OUTCOMES

1. Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

II B. Tech – II Semester

(20BT40531) DATABASE MANAGEMENT SYSTEMS LAB

COURSE OUTCOMES

CO1. Analyze the requirements of a given database problem and design viable ER-Models for implementation of the database.

CO2. Create database schemas, select and apply suitable integrity constraints for querying databases using SQL interface.

CO3. Develop and interpret PL/SQL blocks to centralize database applications for maintainability and reusability.

CO4. Develop database applications for societal applications such as ticket reservation system, employee payroll system using modern tools.

CO5. Work independently and communicate effectively in oral and written forms.

TABLE OF CONTENTS

Title **Page No.**

ABSTRACT

An online food ordering system is software that lets restaurants, coffee shops, or bars accept orders online. It typically allows customers to choose and pay for food, then alerts the kitchen when an order is made. This happens without contact between staff and customers .. Despite the lack of personal interaction, the software provides online customers with the same features someone visiting the restaurant or ordering on the phone would have. This includes access to the entire menu with full customization and varied payment methods such as card, PayPal, or cash on pickup or delivery.

Online ordering platforms should include both a browser-based system so customers can order from their homes or offices, and an app that lets them buy on the move. This gives customers complete ordering flexibility.

The benefits of a good ordering system go way beyond buying food.

Another important consideration is the difference between owning your online ordering system and using a third-party aggregator. Aggregators typically charge huge commissions that eat into your profits. The third-party also stands between your business and the user, making it difficult to build a customer base.

However, with your own system, you have a direct relationship with your customers and don’t pay any commission — potentially saving thousands of dollars each month.

AppInstitute is an all-in-one online food ordering system for restaurants and takeaways. Businesses can implement both a browser-based system and a mobile app to ensure customers have full flexibility in how to order.

Restaurants that use our solution get:

Getting set up via our app builder is easy, and you can be ready to launch in just a few days. Head over to our platform to test out our features for free or get in contact with one of the team to find out more.

1. INTRODUCTION

An online food ordering system is software that lets restaurants, coffee shops, or bars accept orders online. It typically allows customers to choose and pay for food, then alerts the kitchen when an order is made. This happens without contact between staff and customers .. Despite the lack of personal interaction, the software provides online customers with the same features someone visiting the restaurant or ordering on the phone would have. This includes access to the entire menu with full customization and varied payment methods such as card, PayPal, or cash on pickup or delivery.

Online ordering platforms should include both a browser-based system so customers can order from their homes or offices, and an app that lets them buy on the move. This gives customers complete ordering flexibility.

The benefits of a good ordering system go way beyond buying food.

- Run rewards schemes and offer coupons to encourage repeat buys.
- Let users create profiles to save payment information for quick purchases.
- Get access to powerful analytics that give insight into how people buy from your restaurant.
- Give customers easy access to business information such as your address, contact details, and opening and closing hours.
- Reduce barriers to purchase with a messaging center and an FAQ section.

Another important consideration is the difference between owning your online ordering system and using a third-party aggregator. Aggregators typically charge huge commissions that eat into your profits. The third-party also stands between your business and the user, making it difficult to build a customer base.

However, with your own system, you have a direct relationship with your customers and don’t pay any commission — potentially saving thousands of dollars each month.

App Institute is an all-in-one online food ordering system for restaurants and takeaways. Businesses can implement both a browser-based system and a mobile app to ensure customers have full flexibility in how to order.

Restaurants that use our solution get:

- Full menu integration
- Various payment options
- Customer profiles

- Advanced analytics
- Rewards schemes and coupons
- Control over their branding and customer base
- Pay zero commission
- And much more.

Getting set up via our app builder is easy, and you can be ready to launch in just a few days. Head over to our platform to test out our features for free or get in contact with one of the team to find out more.

1.2 Problem Statement

To design and develop a database for the online food delivery system to maintain the records of various restaurants, menu and user's data.

1.3 Objectives

The online food delivery system consists of different types of data such as restaurant names, menu cards, user details, delivery location, delivery boy details, Transaction details. The user login into the website and access all the restaurants in his location and he selects a restaurant, access the menu and then select the items he required and places an order after placing an order he need to mention the location of food delivery and his details and then it asks for the mode of payment it is either through credit card, debit card, net banking, upi, cash on delivery, After selecting the mode of payment ,based on the mode selected it will go through some process and after completion of that process the order will be placed successfully.

2. DATABASE DESIGN

2.1 List of Attributes, Entities and Relationships

The relational database schema for *Online food delivery system* database is as follows:

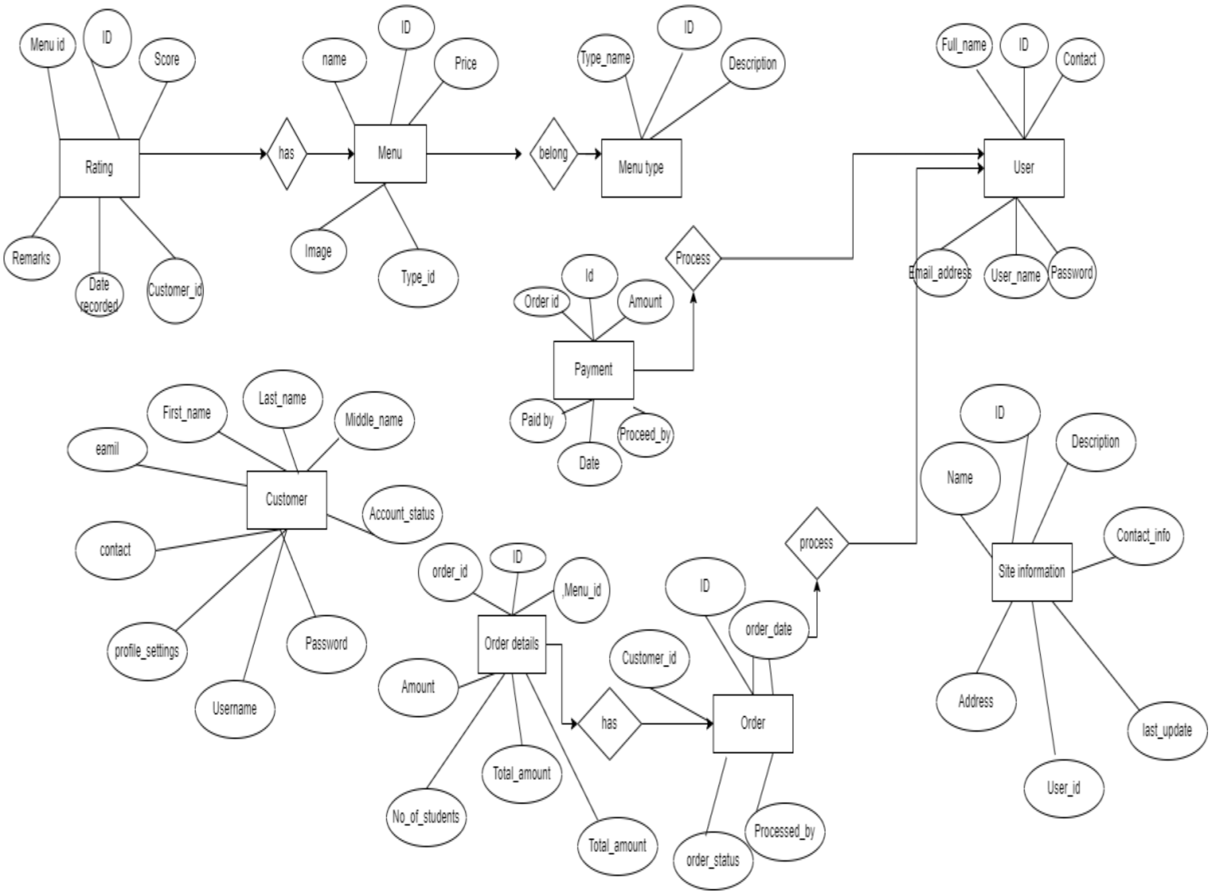
1. **tblmenutype** (menu_type_id int, type_name ,description) ;

2. **tblmenu** (menu_id ,menu_name ,price ,menu_type_id ,ingredients , menu_status)

3. **tblcustomer** (customer_id , customer_first_name , customer_last_name , customer_middle_name , customer_email , customer_phone_number , customer_landline , customer_username , customer_password , account_status) ;

4. **tblorder** (order_id , customer_id , order_date , total_amount , order_status ,processed_by) ;
5. tblorderdetails (order_details_id ,order_id , menu_id ,amount , no_of_serving , total_amount) ;
6. **tblpayment** (payment_id , order_id , amount , paid_by , payment_date , processed_by) ;
7. **tblrating**(rating_id , menu_id , score , remarks , date_recorded , customer_id);
8. **tblsiteinfo** (site_info_id , site_name , description, contact_info , address , last_update , user_id);

List of Relationships: has, belong, process.


Menu id
ID
Score
Rating
Remarks
Date recorded
Customer_id
has
name
ID
Price
Menu
Image
Type_id
belong
Type_name
ID
Description
Menu type
Full_name
ID
Contact
User
Email_address
User_name
Password
Process
Id
Order id
Amount
Payment
Paid by
Date
Proceed_by
First_name
Last_name
Middle_name
eamil
Customer
Account_status
contact
profile_settings
Username
Password
ID
Description
Name
Site information
Contact_info
Address
User_id
last_update
process
order_id
ID
,Menu_id
Order details
Amount
Total_amount
No_of_students
Total_amount
has
ID
order_date
Customer_id
Order
order_status
Processed_by

ER DIAGRAM

3. RELATIONAL MODEL

3.1 Database languages

● A DBMS has appropriate languages and interfaces to express database queries and updates.

● Database languages can be used to read, store and update the data in the database.

1. Data Definition Language:

● DDL stands for Data Definition Language. It is used to define database structure or pattern.

● It is used to create schema, tables, indexes, constraints, etc. in the database.

● Using the DDL statements, you can create the skeleton of the database.

● Data definition language is used to store the information of metadata like the number of tables and schemas, their names, indexes, columns in each table, constraints, etc.

Here are some tasks that come under DDL:

● **Create:** It is used to create objects in the database.

Syntax: CREATE TABLE table_name (column1 datatype, column 2 datatype, column 3 datatype,....);

● **Alter:** It is used to alter the structure of the database.

Syntax: ALTER TABLE table_name MODIFY COLUMN column_name datatype;

● **Drop:** It is used to delete objects from the database.

Syntax: DROP TABLE table_name;

● **Truncate:** It is used to remove all records from a table.

Syntax: TRUNCATE TABLE table_name;

● **Rename:** It is used to rename an object.

Syntax: ALTER TABLE old_table_name RENAME TO new_table_name;

- **Comment:** It is used to comment on the data dictionary.

Syntax: Single line comments start with -- and Multi-line comments starts with
/* and ends with */.

These commands are used to update the database schema that's why they come under Data definition language.

2. Data Manipulation Language

DML stands for Data Manipulation Language. It is used for accessing and manipulating data in a database. It handles user requests.

Here are some tasks that come under DML:

- **Select:** It is used to retrieve data from a database.

Syntax: SELECT column1, column2, … FROM table_name;

- **Insert:** It is used to insert data into a table.

Syntax: INSERT INTO table_name (column1, column2, column3, ...)VALUES (value1, value2, value3, ...);

- **Update:** It is used to update existing data within a table.

Syntax: UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;

- **Delete:** It is used to delete all records from a table.

Syntax: DELETE FROM table_name WHERE condition;

- **Merge:** It performs UPSERT operation, i.e., insert or update operations.

Syntax: MERGE target_table USING source_table ON merge_condition WHEN MATCHED THEN update_statement WHEN NOT MATCHED THEN insert_statement WHEN NOT MATCHED BY SOURCE THEN DELETE;

- **Call:** It is used to call a structured query language or a Java subprogram.

Syntax: CALL procedure_name(parameter[param1, param2, ...])

3.2 Table Description

Following are the tables along with constraints used in the Online food delivery system database.

1.customer table

This table consists of Customer details. The information stored in this

table is: customer_id,

customer_first_name, customer_last_name, customer_middle

name ,customer_phone_number, customer_landline_number, custom

er_username ,customer_password ,account_status.

Constraint: customer_id will be unique for every customer and it is

a primary key.

2.Payment table

this table contains data related to payment section. the info stored in

this table is: payment id, order id, amount, paid by, payment date,

processed by.

Constraint: payment id will be unique for every payment, and it is

primary key in this table.

3.menu type table

This table contains details about type of menu to view in the app. the

info contained in this table is: menu type id, type name, description.

Constraint: menu type id will be unique and it is primary key.

4.menu table

This table contains details of the menu.info is menu id, menu name,

price, menu type, ingredients, menu status.

Constraint: the primary key is menu id and it will be unique.

5.order table

This table contains the details about the order, the info it contains is

Order id, customer id, order date, total amount, order status, processed by.

Constraint: order id is unique and it is primary key for this table.

6.order details table

This table contains info about the details of orders and order details id, order id, menu id, amount, no of serving, total amount.

Constraints: order details id is unique and it is primry key for the table.

7.site info table

This table contains details about the site info. the info is site info id, site name, description, contact info, address, last update, user id.

Constraints: site info id is unique and it is primary key for this table.

8. tbl rating table

This table contains the ratings of food items. The data contained in this table is rating id, menu id, score, remarks, date recorded, customer id.

Constraints:rating id is unique and it is primary key for this table.

3.3 Relational Database Schema

TABLE tblmenutype

Column Name	Data Type	Remarks
menu_type_id	id	PRIMARY KEY
type_name	Varchar(50)	NOT NULL
description	varchar(100)	NOT NULL

Column Name	Data type	Remarks
menu_type_id	id	Primary key
type_name	Varchar2(50)	Not null
description	Varchar2(100)	Not null

TABLE tblmenu

Column Name	Data Type	Remarks
menu_id	Int	PRIMARY KEY
menu_name	Varchar2(100)	NOT NULL
price	Float	NOT NULL
menu_type_id	Int	NOT NULL
ingredients	Varchar2(500)	NOT NULL
menu_status	Int	NOT NULL

TABLE tblcustomer

Column Name	Data Type	Remarks
customer_id	Int	PRIMARY KEY
customer_first_name	Varchar2(30)	NOT NULL
customer_last_name	Varchar2(30)	NOT NULL
customer_middle_name	Varchar2(30)	NOT NULL
customer_email	Varchar2(50)	NOT NULL
customer_phone_number	Varchar2(15)	NOT NULL
customer_landline	Varchar2(15)	NOT NULL
customer_username	Varchar2(30)	NOT NULL
customer_password	Varchar2(30)	NOT NULL
account_status	int	NOT NULL

TABLE tblorder

Column Name	Data Type	Remarks
order_id	int	PRIMARY KEY
customer_id	int	NOT NULL
order_date	date	NOT NULL

total_amount	float	NOT NULL
order_status	int	NOT NULL
processed_by	int	NOT NULL

TABLE tblorderdetails

Column Name	Data Type	Remarks
order_details_id	int	PRIMARY KEY
order_id	int	NOT NULL
menu_id	int	NOT NULL
amount	float	NOT NULL
no_of_serving	int	NOT NULL
total_amount	float	NOT NULL

TABLE tblpayment

Column Name	Data Type	Remarks
payment_id	Int	PRIMARY KEY
order_id	Int	NOT NULL
amount	Float	NOT NULL
paid_by	Varchar2(50)	NOT NULL
payment_date	date	NOT NULL
processed_by	Int	NOT NULL

TABLE tblrating

Column Name	Data Type	Remarks
rating_id	int	PRIMARY KEY
menu_id	int	NOT NULL
score	int	NOT NULL
remarks	Varchar2(100)	NOT NULL
date_recorded	Date	NOT NULL
customer_id	Int	NOT NULL

TABLE tblsiteinfo

Column Name	Data Type	Remarks
site_info_id	Int	PRIMARY KEY
site_name	Varchar2(30)	NOT NULL
description	Varchar2(100)	NOT NULL
contact_info	Varchar2(15)	NOT NULL
address	Varchar2(100)	NOT NULL
last_update	date	NOT NULL
user_id	int	NOT NULL

3.4 Relational Queries

● Creation of database Online food delivery system:

```
CREATE database Online food delivery system
```

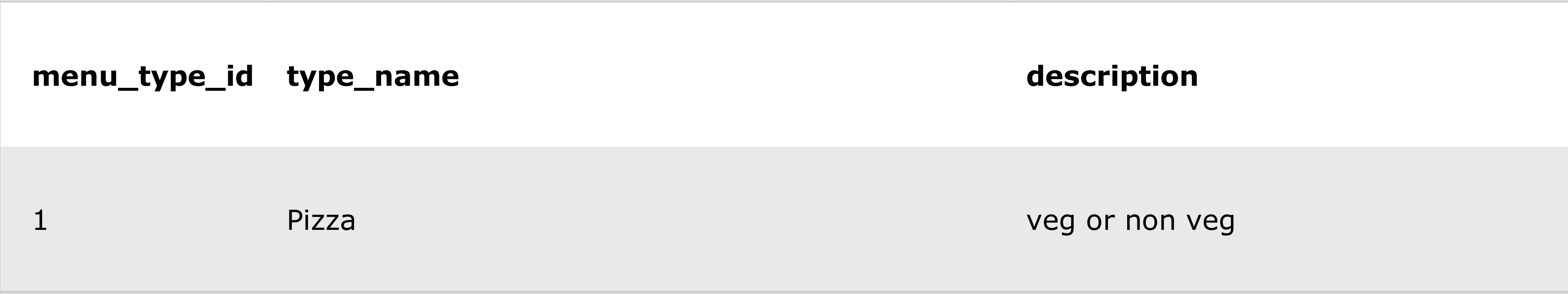
● Create table tblmenutype:

```
CREATE TABLE tblmenutype (
 menu_type_id int NOT NULL ,
 type_name varchar(50) NOT NULL,
 description varchar(100) NOT NULL,
 PRIMARY KEY (menu_type_id)
 ) ;
```

Insertion of values into tblmenutype table:

```
insert into tblmenutype values(1,'pizza','veg or non veg');
insert into tblmenutype values(2,'biryani','veg or non veg');
insert into tblmenutype values(5,'starters','veg or non veg');
insert into tblmenutype values(5,'burgers','veg or non veg');
insert into tblmenutype values(5,'curry','veg or non veg');
```

Output: SELECT * FROM [tblmenutype]



menu_type_id	type_name	description
1	Pizza	veg or non veg

2	Biryani	veg or non veg
3	Curry	veg or non veg
4	Starters	veg or non veg
5	Burgers	veg or non veg

- **Create table tblmenu:**

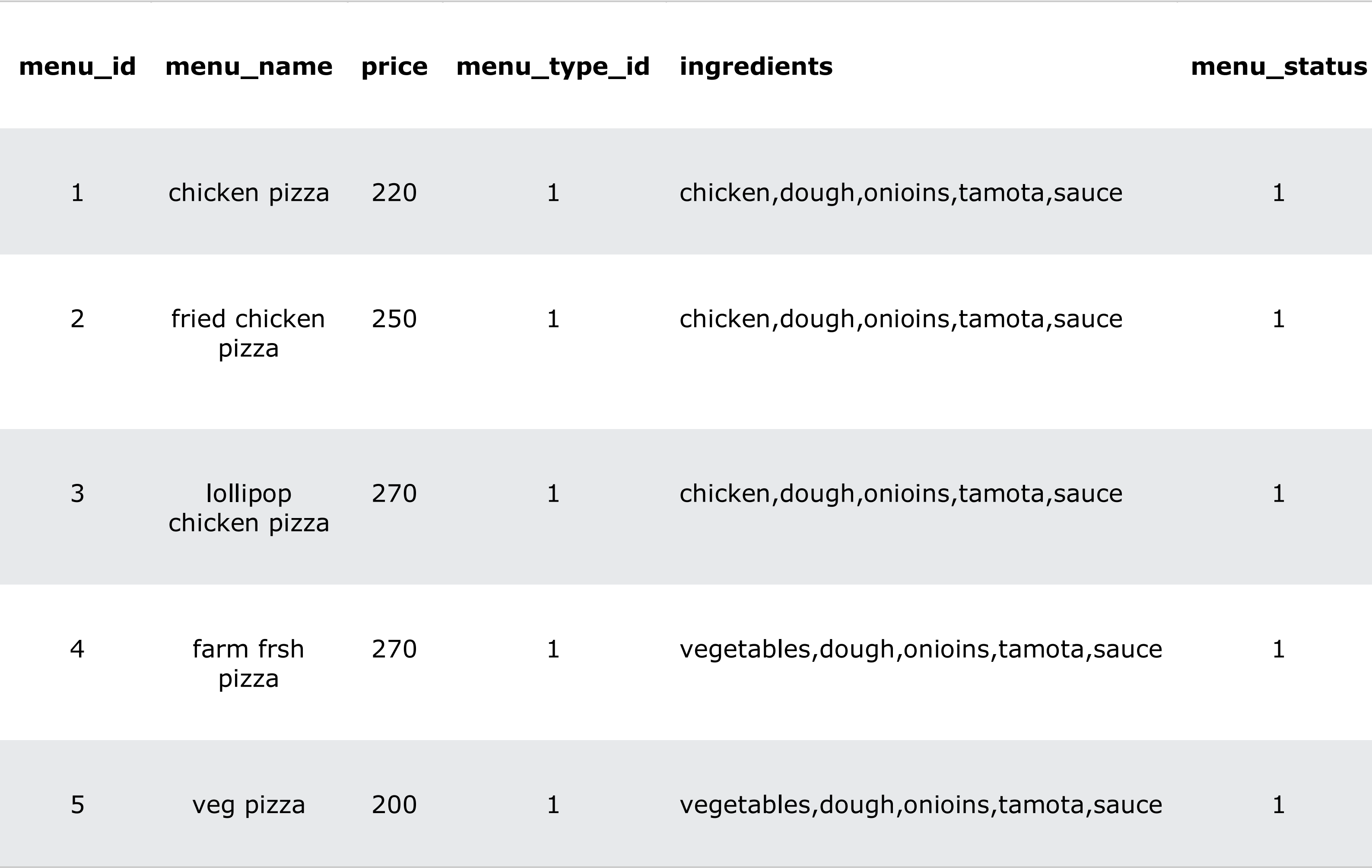
```
CREATE TABLE tblmenu (

menu_id int NOT NULL ,

menu_name varchar(100) NOT NULL,

price float NOT NULL,

menu_type_id int NOT NULL,

ingredients varchar(500) NOT NULL,

menu_status int NOT NULL,

PRIMARY KEY (menu_id)

) ;
```

Insertion of values into tblmenu table:

```
Insert into tblmenu values(1,’chicken
pizzza’,220,1,’chicken,dough,onions,tamota,sauce’,1);
```

```
Insert into tblmenu values(2,’fried chicken pizzza’,250,1,’chicken,dough,onions,tamota,sauce’,1);

Insert into tblmenu values(3,’lollipop chicken pizzza’,270,1,’chicken,dough,onions,tamota,sauce’,1);

Insert into tblmenu values(4,’farm fresh pizzza’,270,1,’vegetables,dough,onions,tamota,sauce’,1);\

Insert into tblmenu values(5,’veg pizzza’,200,1,’vegtables,dough,onions,tamota,sauce’,1);
```

Output: SELECT * FROM [tblmenu]



menu_id	menu_name	price	menu_type_id	ingredients	menu_status
1	chicken pizza	220	1	chicken,dough,onioins,tamota,sauce	1
2	fried chicken pizza	250	1	chicken,dough,onioins,tamota,sauce	1
3	lollipop chicken pizza	270	1	chicken,dough,onioins,tamota,sauce	1
4	farm frsh pizza	270	1	vegetables,dough,onioins,tamota,sauce	1
5	veg pizza	200	1	vegetables,dough,onioins,tamota,sauce	1

Output:

- **Create table tblcustomer :**

```
CREATE TABLE tblcustomer (

customer_id int NOT NULL ,

customer_first_name varchar(30) NOT NULL,

customer_last_name varchar(30) NOT NULL,

customer_middle_name varchar(30) NOT NULL,

customer_email varchar(50) NOT NULL,

customer_phone_number varchar(15) NOT NULL,

customer_landline varchar(15 ) NOT NULL,

customer_username varchar(30) NOT NULL,

customer_password varchar(30) NOT NULL,

account_status int NOT NULL,

PRIMARY KEY (customer_id)

) ;
```

Insertion of values into tblcustomer table:

```
insert into tblcustomer
values(1,'a','b','c','abc@gmail.com',9689499391,9689499391,'abc','abc',1);

insert into tblcustomer
values(2,'b','c','d','bcd@gmail.com',9689499392,9689499392,'bcd','bcd',1);

insert into tblcustomer
values(3,'c','d','e','cde@gmail.com',9689499393,9689499393,'cde','cde',1);

insert into tblcustomer
values(4,'d','e','f','def@gmail.com',9689499394,9689499394,'def','def',1);

insert into tblcustomer
values(5,'e','f','g','efg@gmail.com',9689499395,9689499395,'efg','efg',1);
```

Output: SELECT * FROM [tblcustomer]

id	F_name	L_name	M_name	gmail	Phone	landline	User_name	passwrd	status
1	a	b	c	abc@gmail.com	9689499391	9689499391	abc	abc	1
2	b	c	d	bcd@gmail.com	9689499392	9689499392	bcd	bcd	1
3	c	d	e	cde@gmail.com	9689499393	9689499393	cde	cde	1
4	d	e	f	def@gmail.com	9689499394	9689499394	def	def	1
5	e	f	g	efg@gmail.com	9689499395	9689499395	efg	efg	1

- **Create table tblorder :**

```
CREATE TABLE tblorder (

order_id int NOT NULL ,

customer_id int NOT NULL,

order_date date NOT NULL,

total_amount float NOT NULL,

order_status int NOT NULL,

processed_by int NOT NULL,
```

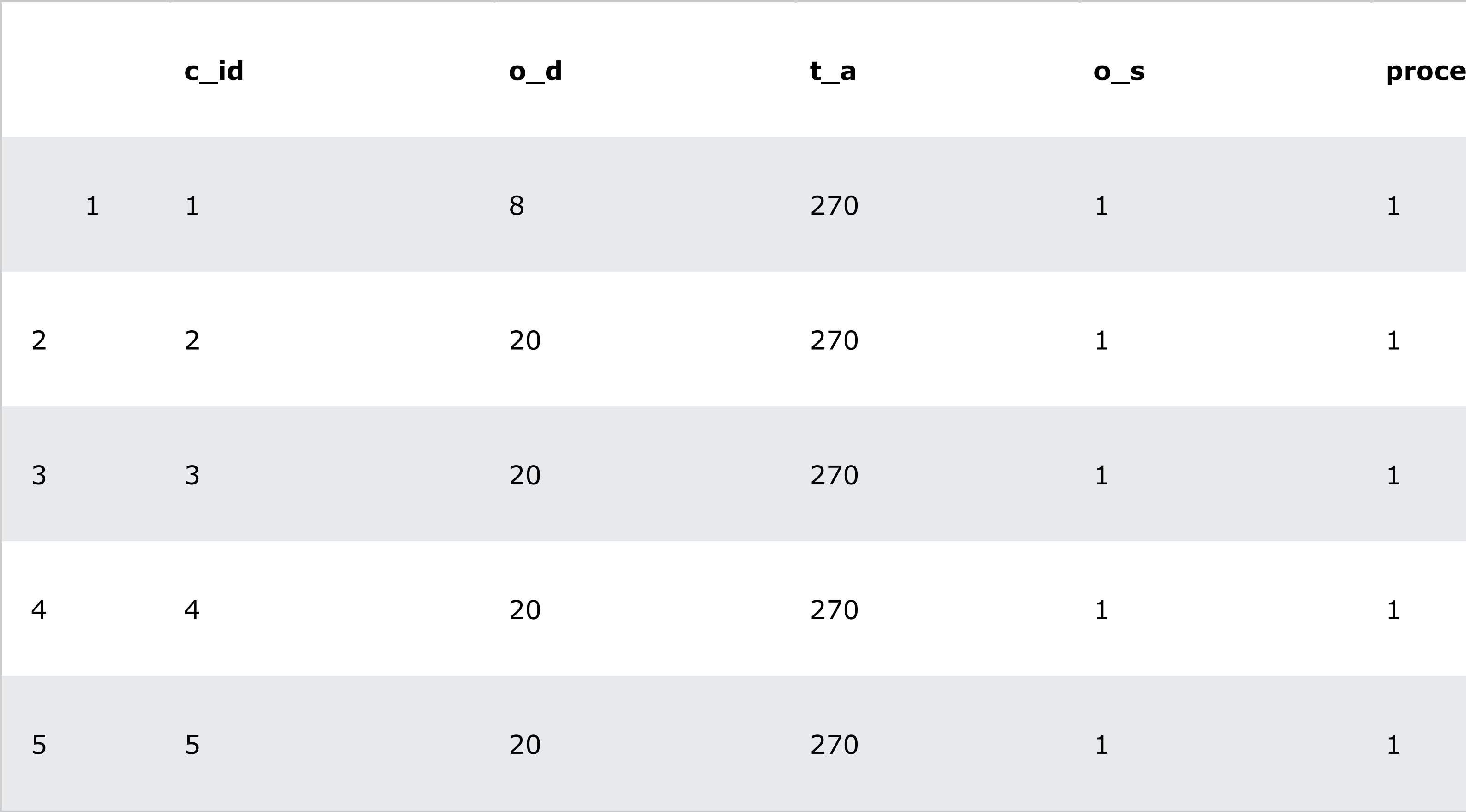
PRIMARY KEY (order_id)

) ;

Insertion of values into tblorder table:

```
insert into tblorder values(1,1,20 ,270,1,1);

insert into tblorder values(2,2,20,270,1,1);

insert into tblorder values(3,3,20,270,1,1);

 insert into tblorder values(4,4,20,270,1,1);

insert into tblorder values(5,5,20,270,1,1);
```

Output: SELECT * FROM [tblorder]



	c_id	o_d	t_a	o_s	proce
1	1	8	270	1	1
2	2	20	270	1	1
3	3	20	270	1	1
4	4	20	270	1	1
5	5	20	270	1	1

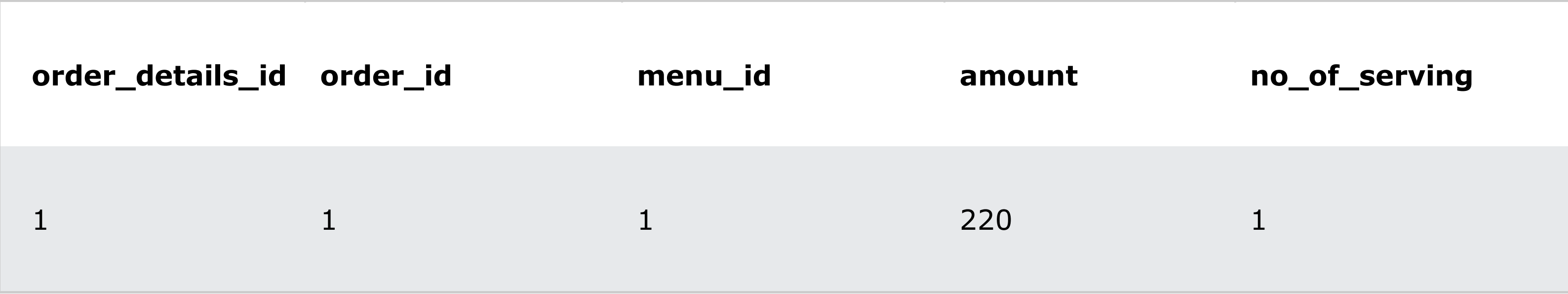
- **Create table tblorderdetails :**

```
CREATE TABLE tblorderdetails (

order_details_id int NOT NULL ,

order_id int NOT NULL,

menu_id int NOT NULL,

amount float NOT NULL,


no_of_serving int NOT NULL,

total_amount float NOT NULL,

PRIMARY KEY (order_details_id)

) ;
```

Insertion of values into tblorderdetails table:

```
insert into tblorderdetails values(1,1,1,220,1,220);

insert into tblorderdetails values(2,2,1,220,1,220);

insert into tblorderdetails values(3,3,1,220,1,220);

insert into tblorderdetails values(4,4,1,220,1,220);

insert into tblorderdetails values(5,5,1,220,1,220);
```

Output: SELECT * FROM [tblorderdetails]



order_details_id	order_id	menu_id	amount	no_of_serving
1	1	1	220	1

2	2	1	220	1
3	3	1	220	1
4	4	1	220	1
5	5	1	220	1

- **Create table tblpayment :**

CREATE TABLE tblpayment (

payment_id int NOT NULL ,

order_id int NOT NULL,

amount float NOT NULL,

paid_by varchar(50) NOT NULL,

payment_date date NOT NULL,

processed_by int NOT NULL,

PRIMARY KEY (payment_id)

) ;

Insertion of values into tblpayment table:

insert into tblpayment values(1,1,220,'abc',20,'me');

insert into tblpayment values(2,2,220,'bcd',20,'me');

insert into tblpayment values(3,3,220,'cde',20,'me');

```
insert into tblpayment values(4,4,220,'def',20,'me');

insert into tblpayment values(5,5,220,'efg',20,'me');
```

Output: SELECT * FROM [tblpayment]

payment_id	order_id	amount	paid_by
1	1	220	abc
2	2	220	bcd
3	3	220	cde
4	4	220	def
5	5	220	efg

- **Create table tblrating:**

```
CREATE TABLE tblrating(

rating_id int NOT NULL ,

menu_id int NOT NULL,
score int NOT NULL,

remarks varchar(100) NOT NULL,

date_recorded date NOT NULL,

customer_id int NOT NULL,

PRIMARY KEY (rating_id)

);
```

Insertion of values into tblrating table:

```
insert into tblrating values(1,1,5,'ntg',20,1);

insert into tblrating values(2,2,5,'ntg',20,2);

insert into tblrating values(3,3,5,'ntg',20,3);

insert into tblrating values(4,4,5,'ntg',20,4);

insert into tblrating values(5,5,5,'ntg',20,5);
```

Output: SELECT * FROM [tblrating]

`rating_id	menu_id	score	remarks
1	1	5	ntg
2	2	5	ntg
3	3	5	ntg

4	4	5	ntg
5	5	5	ntg

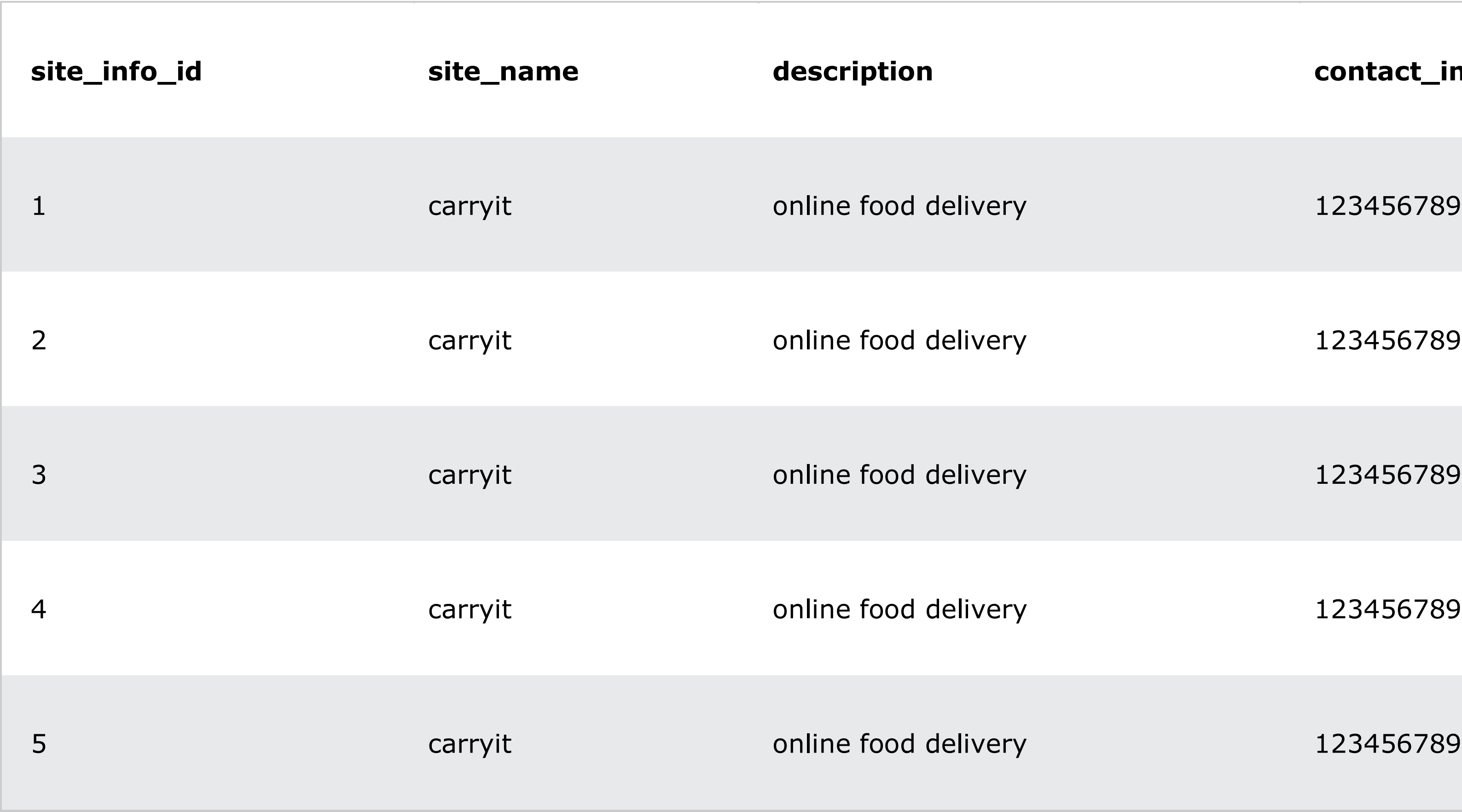
- **Create table tblsiteinfo:**

```
CREATE TABLE tblsiteinfo (

site_info_id int NOT NULL,

site_name varchar(30) NOT NULL,

description varchar(100) NOT NULL,

contact_info varchar(15) NOT NULL,

address varchar(100) NOT NULL,

last_update date NOT NULL,

user_id int NOT NULL,

PRIMARY KEY (site_info_id)

);

);
```

Insertion of values into tblsiteinfo table:

```
insert into tblsiteinfo values(1,'carryit','online food delivery',1234567890,'earth',20,1);

insert into tblsiteinfo values(2,'carryit','online food delivery',1234567890,'earth',20,2);

insert into tblsiteinfo values(3,'carryit','online food delivery',1234567890,'earth',20,3);
insert into tblsiteinfo values(4,'carryit','online food
delivery',1234567890,'earth',20,4);

insert into tblsiteinfo values(5,'carryit','online food
delivery',1234567890,'earth',20,5);
```

Output: SELECT * FROM [tblsiteinfo]



site_info_id	site_name	description	contact_in
1	carryit	online food delivery	123456789
2	carryit	online food delivery	123456789
3	carryit	online food delivery	123456789
4	carryit	online food delivery	123456789
5	carryit	online food delivery	123456789

SQL QUERIES:

1) Find the number of items available in the menu?

SELECT count(menu_id) FROM tblmenu;

count(menu_id)
5

2) Find the menu_names of the menu table ?

SELECT menu_name FROM [tblmenu]

menu_name
chicken pizza
fried chicken pizza
lollipop chicken pizza
farm frsh pizza
veg pizza

3) Minimum price items in the menu?

SELECT min(price) FROM [tblmenu]

min(price)
200

4) Maximum price items in the menu?

SELECT max(price) FROM [tblmenu]

max(price)
270

5) Sum of ordered items?

6) What items ordered by many peoples?

7) What is the maximum and minimum rating of the hotel?

SELECT max(score),min(score) FROM [tblrating]

max(score)	min(
5	5

8) What is the average rating of the hotel?

SELECT avg(score) FROM [tblrating]

avg(score)
5

9) Retrieve the item name from the menu?

menu_name
chicken pizza
fried chicken pizza
lollipop chicken pizza
farm frsh pizza

veg pizza

SELECT menu_name FROM [tblmenu]

10) Retrieve the customer names, customer ID from customer?

SELECT customer_username,customer_id FROM [tblcustomer]

customer_username
abc
bcd
cde
def
efg

11) Retrieve the customer name ordered item, payment price?

SELECT customer_username,customer_id FROM [tblcustomer]

customer_username

abc
bcd
cde
def
efg

12) Orderd items.

SELECT distinct m.menu_name FROM tblmenu m,tblorderdetails t where m.menu_id=t.menu_id;

menu_name
chicken pizza

13) Number of categories in menu.

SELECT count(menu_type_id) FROM [tblmenutype]

count(menu_type_id)

5

14) Print bill.

SELECT c.customer_username,o.amount from tblcustomer c,tblorderdetails o where c.customer_id=o.order_id;

customer_username
abc
bcd
cde
def
efg

15) Display the Payment ID of a customer.

SELECT c.customer_username,o.order_id from tblcustomer c,tblorderdetails o where c.customer_id=o.order_id;

customer_username

abc
bcd
cde
def
efg

16) Count the available item in menu.

SELECT menu_name FROM [tblmenu]

menu_name
chicken pizza
fried chicken pizza
lollipop chicken pizza
farm frsh pizza

veg pizza

17)Count the category item in menu.

SELECT type_name FROM [tblmenutype]

type_name
pizza
biryani
curry
starters
burgers

18)Display the food item.

SELECT type_name FROM [tblmenutype]

type_name
pizza

biryani
curry
starters
burgers

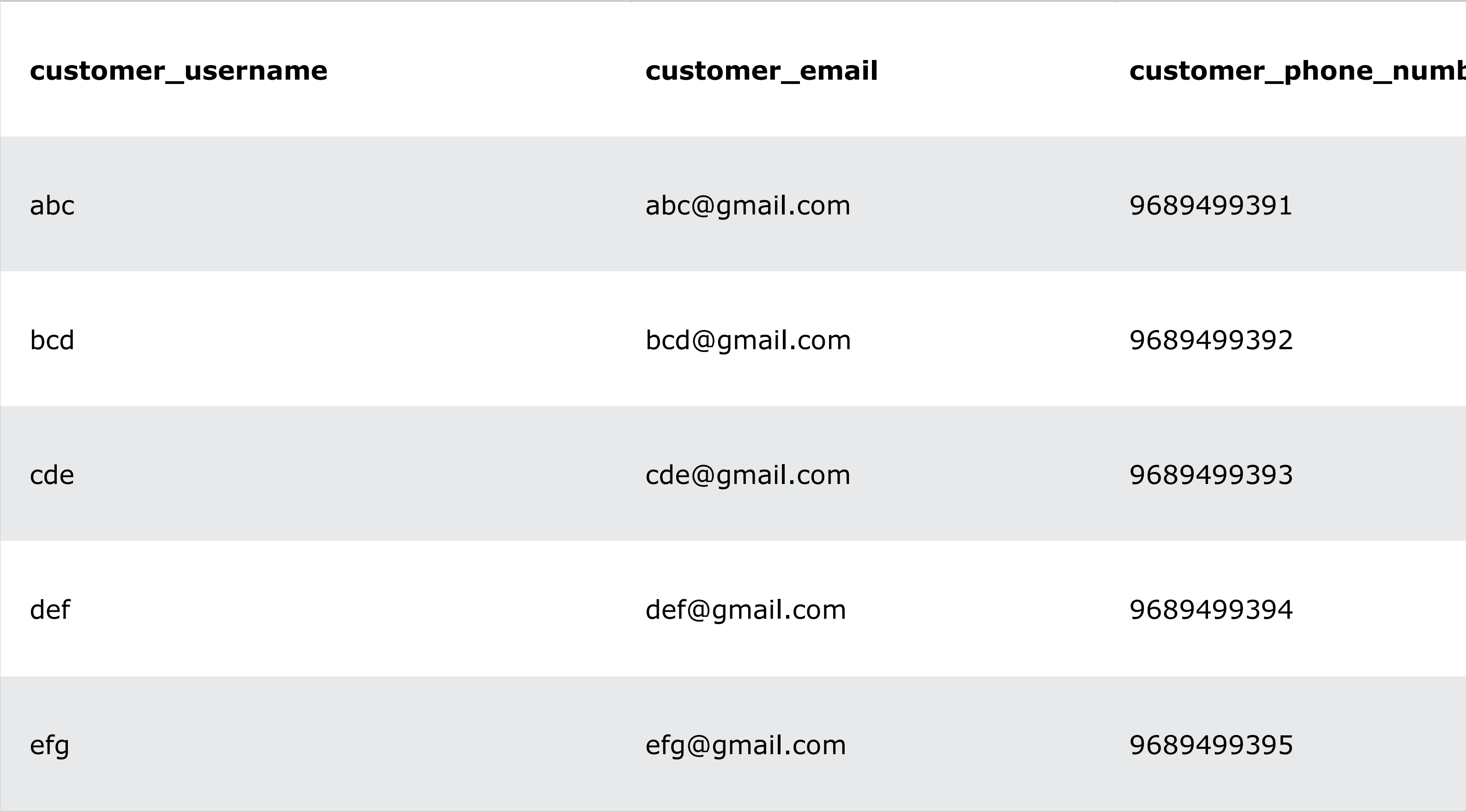
19) List the ingrediants of the food item.

SELECT menu_name,ingredients FROM [tblmenu]

menu_name	ingredients
chicken pizza	chicken,dough,onioins,tamota,sauce
fried chicken pizza	chicken,dough,onioins,tamota,sauce
lollipop chicken pizza	chicken,dough,onioins,tamota,sauce
farm frsh pizza	vegetables,dough,onioins,tamota,sauce
veg pizza	vegetables,dough,onioins,tamota,sauce

20) Details of the customer using payment ID.

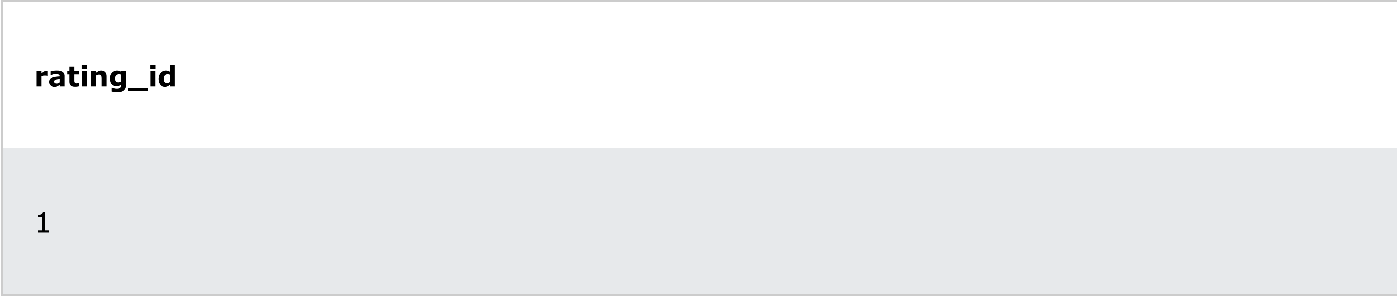
SELECT c.customer_username,c.customer_email,c.customer_phone_number,c.customer_password FROM tblcustomer c, tblpayment p,tblorder o where c.customer_id=o.order_id and o.order_id=p.order_id;



customer_username	customer_email	customer_phone_numb
abc	abc@gmail.com	9689499391
bcd	bcd@gmail.com	9689499392
cde	cde@gmail.com	9689499393
def	def@gmail.com	9689499394
efg	efg@gmail.com	9689499395

21) Rating of the particular item.

SELECT rating_id,score FROM [tblrating]



rating_id
1

2
3
4
5

22) Display site information.

SELECT * FROM [tblsiteinfo]

site_info_id	site_name	description	contact_info
1	carryit	online food delivery	1234567890
2	carryit	online food delivery	1234567890
3	carryit	online food delivery	1234567890
4	carryit	online food delivery	1234567890
5	carryit	online food delivery	1234567890

23) Add the new items into the menu.

24) What is the most demanded item?

25) Which item is not ordered at all?

26) What amount of bussiness is made on daily basis?

27) Display the all the table.

28) Display the order details ?

29) Count the number of customers ordering the food items.

30) Average of the prices.

4. CONCLUSION AND FUTURE WORK

4.1 Conclusion

The Online food delivery system was implemented keeping in mind to create a tool for all the restaurants which they would integrate in their existing customers and new attract new customers. Allow a user to create an account through registering on the website Allow to view account history, bill, menu Bill should be deducted when the payment is made Allow one to been successfully placed The system should a database that keeps the users details It should be able to create reports.

4.2 Future Work

Databases are used for storing, maintaining and accessing any sort of data. They collect information on people, places or things. That information is gathered in one place so that it can be observed and analyzed. Databases can be thought of as an organized collection of information and this helps us in the creation of a website or a mobile app where it can be accessed by viewers globally.

An online food ordering system is developed where the customers can make an order for the food and avoid the hassles of waiting for the order to be taken by the waiter. Using the application, the end users register online, read the E-menu card and select the food from the e-menu card to order food online. Once the customer selects the required food item the chef will be able to see the results on the screen and start processing the food. This application nullifies the need of a waiter or reduces the workload of the waiter. The advantage is that in a crowded restaurant there will be chances that the waiters are overloaded with orders and they are unable to meet the requirements of the customer in a satisfactory manner. Therefore by using this application, the users can directly place the order for food to the chef online.