IE 6700: Data Management for Analytics

VirtuChef: Your Cloud Kitchen Experience

Project Report

GROUP - 3

Samyuktha Kapoor
Karthick Sriram Manimaran

857-867-1929 (Samyuktha Kapoor)
857-654-8336 (Karthick Sriram Manimaran)

rajeshkapoor.s@northeastern.edu
manimaran.k@northeastern.edu

Signature of Student 1: Samyuktha Kapoor Rajesh Kapoor

Signature of Student 2: Karthick Sriram Manimaran

Submission Date: 04/21/2024

VirtuChef: Your Cloud Kitchen Experience

I. Introduction:

In our fast-paced world, where time-saving technology paradoxically leaves us with less time for basic needs, like enjoying a good meal, the simplicity of ordering takeout or delivery from a restaurant often comes to the rescue. Seeking convenience and hot, delicious dishes, customers overlook the intricate complexities behind the takeout process. Most restaurants deploy tracking systems for efficient order and delivery management. However, the culinary landscape is evolving with the rise of Cloud Kitchens, virtual spaces where chefs operate without a physical restaurant. This innovative model introduces unique challenges and opportunities, necessitating the application of data management principles to streamline operations, enhance customer experiences, and optimize end-to-end processes. Cloud Kitchens are especially beneficial for busy professionals, students, and instances when guests are at home or health takes precedence. Unlike traditional takeout, Cloud Kitchens offer seamless solutions, with assigned delivery ensuring meals are delivered directly, eliminating the need for customers to pick up food. The core challenge lies in refining and improving the overall functioning of Cloud Kitchens, ensuring they operate smoothly, provide better customer experiences, and make the most of data-driven insights in the swiftly changing culinary industry.

Project Theory:

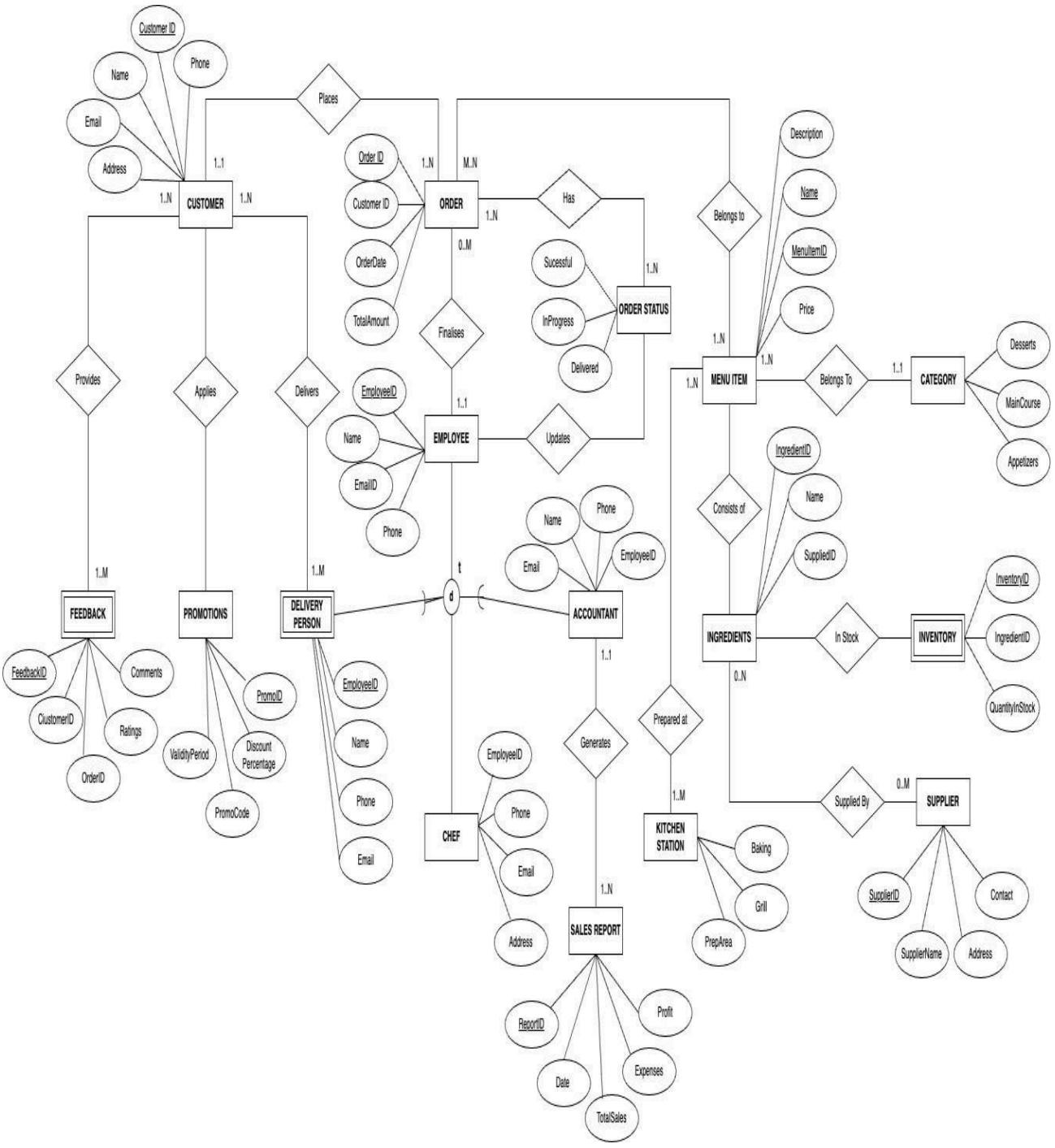
Our project strategically revolutionizes Cloud Kitchen operations by managing entities such as chefs, delivery personnel, customers, suppliers and more. Using robust data management, we streamline the entire lifecycle from ingredient procurement to order fulfillment. By harnessing data-driven insights, our goal is to refine decision-making, allocate resources effectively, and enhance overall customer satisfaction in the dynamic Cloud Kitchen environment.

The project's scope involves the comprehensive integration of entities into a sophisticated Cloud Kitchen database system. This integration aims to enhance operational workflows, enrich customer engagement, optimize order processing, and maximize resource utilization. The resulting structured database not only streamlines internal processes but also acts as a catalyst, enhancing the overall customer experience. Through these advancements, our project aspires to position the Cloud Kitchen as an industry leader in the competitive culinary landscape, setting new standards in the evolving industry.

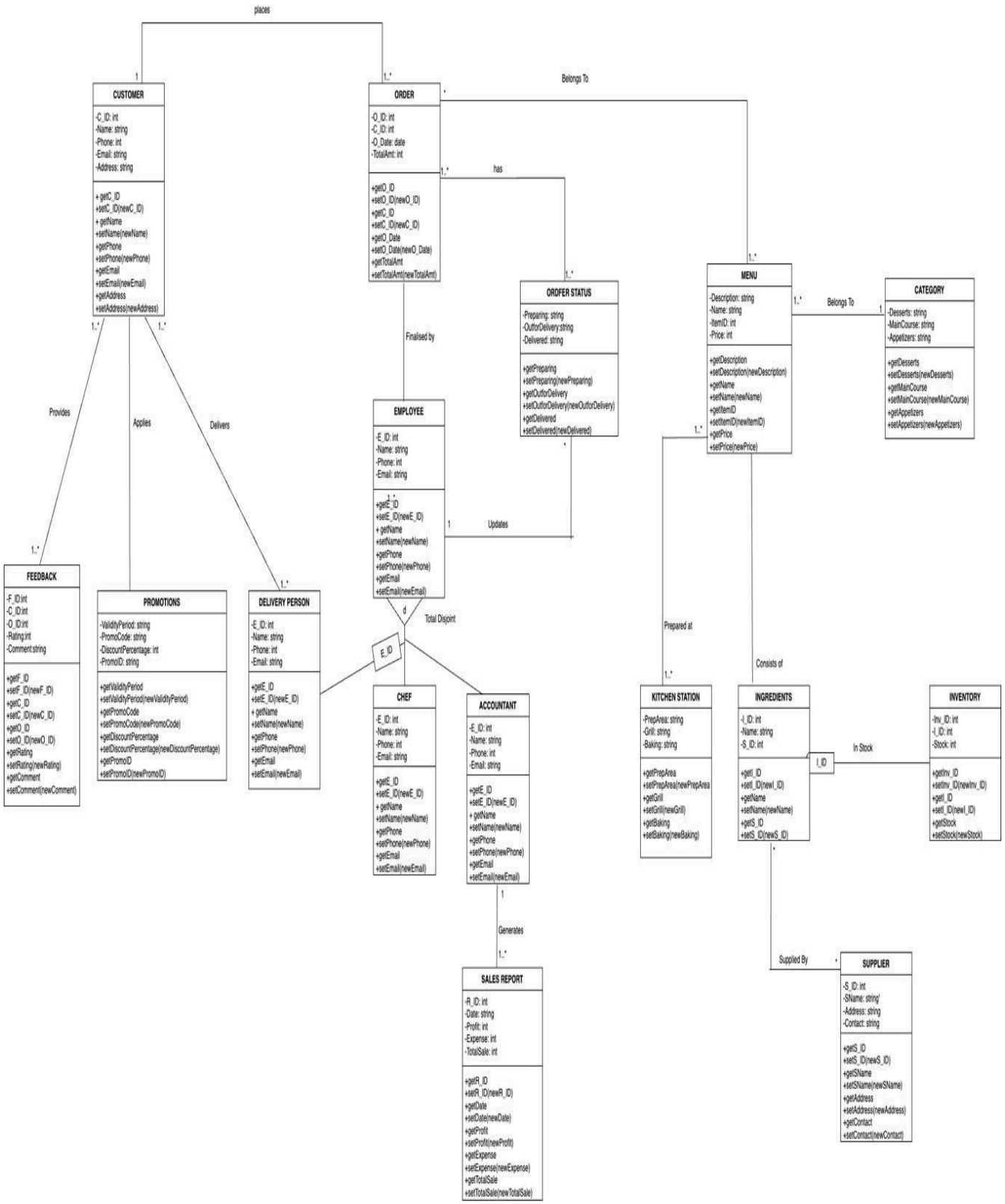
The strategic integration of data management principles within the project ensures the efficient orchestration of Cloud Kitchen entities, creating an environment characterized by culinary excellence and operational efficiency. Our project stands as a testament to the transformative power of data, shaping the future of Cloud Kitchens and exceeding the expectations of a discerning customer base.

II. Conceptual Data Modelling:

1. EER Diagram:



2. UML Diagram:



III. Mapping Conceptual model to Relational model:

(PK - Primary Key, **FK -** Foreign Key**)**

CustomerID: (PK: CustomerID, Name, Phone, Email, Address)

OrderID: (PK: OrderID, FK: CustomerID, OrderDate, TotalAmount)

EmployeeID: (PK: EmployeeID, Name, Phone, Email)

FeedbackID: (PK: FeedbackID, FK: CustomerID, FK: OrderID, Rating, Comments)

PromoID: (PK: PromoID, PromoCode, ValidityPeriod, DiscountPercentage)

DeliveryPersonID: (PK: EmployeeID, FK: EmployeeID, Name, Phone, Email)

ChefID: (PK: EmployeeID, FK: EmployeeID, Name, Phone, Email)

AccountantID: (PK: EmployeeID, FK: EmployeeID, Name, Phone, Email)

OrderStatusID: (PK: StatusID, StatusName)

SalesReportID: (PK: ReportID, Date, Profit, Expenses, TotalSales)

MenuItemID: (PK: MenuItemID, Name, Description, Price)

CategoryID: (PK: CategoryID, CategoryName)

SupplierID: (PK: SupplierID, SupplierName, Address, Contact)

IngredientID: (PK: IngredientID, FK: SupplierID, Name)

InventoryID: (PK: InventoryID, FK: IngredientID, QuantityInStock)

KitchenStationID: (PK: StationID, StationName)

IV. Implementation of Relational Model via SQL and NoSQL

SQL IMPLEMENTATION

QUERY 1: Simple Query

Retrieve all customers with their details.

SELECT * FROM Customer;

Explanation: This query fetches all the information available for every customer in the Customer table.

C_ID	Name	Phone	Email	Address
103	Michael	9012342098	michael@example.com	789 Oak St
104	Sarah	0123453209	sarah@example.com	101 Pine St
105	David	1234564310	david@example.com	111 Cedar St
106	Jessica	2345675421	jessica@example.com	222 Maple St
107	Daniel	3456786532	daniel@example.com	333 Birch St
108	Sophia	4567897643	sophia@example.com	444 Elm St
109	Matthew	5678908754	matthew@example.com	555 Oak St
110	Emma	6789019865	emma@example.com	666 Pine St
111	Christo…	7890130976	christopher@example.…	777 Cedar St
112	Olivia	8901242087	olivia@example.com	888 Maple St
113	Andrew	9012353198	andrew@example.com	999 Birch St
114	Isabella	0123464209	isabella@example.com	1010 Elm St
115	James	1234575310	james@example.com	1111 Oak St
116	Ava	2345686421	ava@example.com	1212 Pine St
117	Alexan…	3456797532	alexander@example.c…	1313 Cedar…

QUERY 2: Aggregate

Count the number of orders for each status.

SELECT Status_ID, COUNT(*) AS TotalOrders FROM Orders
GROUP BY Status_ID;

Explanation: This query calculates how many orders there are for each type of status (like confirmed, preparing, etc.) and displays the count next to each status ID.

Status_ID	TotalOrders
1	25
2	25
3	24
4	25

QUERY 3: Inner Join

Inner join between Customers and Orders to show customer names with their orders' total amounts

SELECT Customer.Name, Orders.TotalAmount FROM Customer
JOIN Orders ON Customer.C_ID = Orders.C_ID;

Explanation: This query links (or joins) customers with their orders to display each customer's name along with how much their order was worth.

Name	TotalAmount
John	10
Emily	12
Michael	15
Sarah	20
David	11
Jessica	13
Daniel	16
Sophia	21
Matthew	12

QUERY 4: Outer Join

Left outer join to find all customers and their order statuses, including those without orders. SELECT Customer.Name, OrderStatus.Status

FROM Customer

LEFT JOIN Orders ON Customer.C_ID = Orders.C_ID

LEFT JOIN OrderStatus ON Orders.Status_ID = OrderStatus.Status_ID;

Explanation: This query lists all customers, even those who haven't placed any orders, along with the status of any orders they might have placed.

Name	Status
John	Order confirmed
Emily	Preparing
Michael	Picked up
Sarah	Delivered
David	Order confirmed
Jessica	Preparing
Daniel	Picked up
Sophia	Delivered
Matthew	Order confirmed

QUERY 5: Nested Query
List customers who have placed orders above the average order amount.

```
SELECT Name FROM Customer
WHERE C_ID IN (SELECT C_ID FROM Orders WHERE TotalAmount > (SELECT
AVG(TotalAmount) FROM Orders));
```

Explanation: This query identifies customers who have spent more than the average spent across all orders, highlighting high-spending customers.

Name
Ethan
Chloe
Gabriel
Nathan
Zack
Carla

QUERY 6: Correlated Query
Find each order's amount as compared to the average amount of all orders of the same status.

```
SELECT O_ID, TotalAmount,
    (SELECT AVG(TotalAmount) FROM Orders O2 WHERE O1.Status_ID = O2.Status_ID) AS
AvgStatusAmount
FROM Orders O1;
```

Explanation: This query checks how much each order cost in comparison to the average cost of orders that have the same status.

O_ID	TotalAmount	AvgStatusAmou...
1	10	24.2
2	12	26
3	15	27.5
4	20	31.76
5	11	24.2
6	13	26
7	16	27.5
8	21	31.76

QUERY 7: >=ALL/>ANY/EXISTS/NOT EXISTS
Find customers who have spent more than any customer named 'John'.

```
SELECT Name FROM Customer WHERE C_ID >= ALL (
    SELECT C_ID FROM Orders WHERE TotalAmount > ANY (
      SELECT TotalAmount FROM Orders JOIN Customer ON Customer.C_ID = Orders.C_ID
WHERE Name = 'John'
    )
);
```

Explanation: This query finds customers whose total order amount is greater than the order amounts of any customer named John.

Name
Isabel

QUERY 8: Set Operations (Union)
List all unique customer and employee names together. SELECT Name FROM Customer
UNION
SELECT Name FROM Employee;

Explanation: This query combines and displays a list of unique names from both the Customer and Employee tables.

Name
John
Emily
Michael
Sarah
David
Jessica
Daniel

QUERY 9: Subqueries in Select and From
Subquery in SELECT to find the number of orders for each customer.

SELECT Name, (SELECT COUNT(*) FROM Orders WHERE Orders.C_ID =
Customer.C_ID) AS OrderCount

FROM Customer;

Explanation: This query shows each customer's name along with the total number of orders they have placed.

Name	OrderCount
John	1
Emily	1
Michael	1
Sarah	1
David	1
Jessica	1
Daniel	1

QUERY 10: Subquery in FROM to list customers and their total spent.

```
SELECT C.Name, TotalSpent FROM Customer C JOIN (
    SELECT C_ID, SUM(TotalAmount) AS TotalSpent FROM Orders
    GROUP BY C_ID
) AS Spending ON C.C_ID = Spending.C_ID;
```

Explanation: This query lists customers along with the total amount they have spent on orders, providing insight into customer spending behavior.

Name	TotalSpent
John	10
Emily	12
Michael	15
Sarah	20
David	11
Jessica	13

NO SQL IMPLEMENTATION

QUERY 1: Simple Query (Find All Documents in a Collection)

a. To retrieve all documents from the employees collection:

db.Employee.find({});

Explanation: This MongoDB command retrieves all documents within the employees collection, similar to a SELECT * in SQL.

```
> db.Employee.find({});
< {
    _id: ObjectId('661b01b03caf3fa6a60570b5'),
    E_ID: 15,
    Name: 'Isabella Martin',
    Phone: 5678901234,
    Email: 'isabella.martin@example.com',
    Type: 'Chef'
  }
  {
    _id: ObjectId('661b01b03caf3fa6a60570b0'),
    E_ID: 10,
    Name: 'Olivia Anderson',
    Phone: 123456789,
    Email: 'olivia.anderson@example.com',
    Type: 'Chef'
  }
```

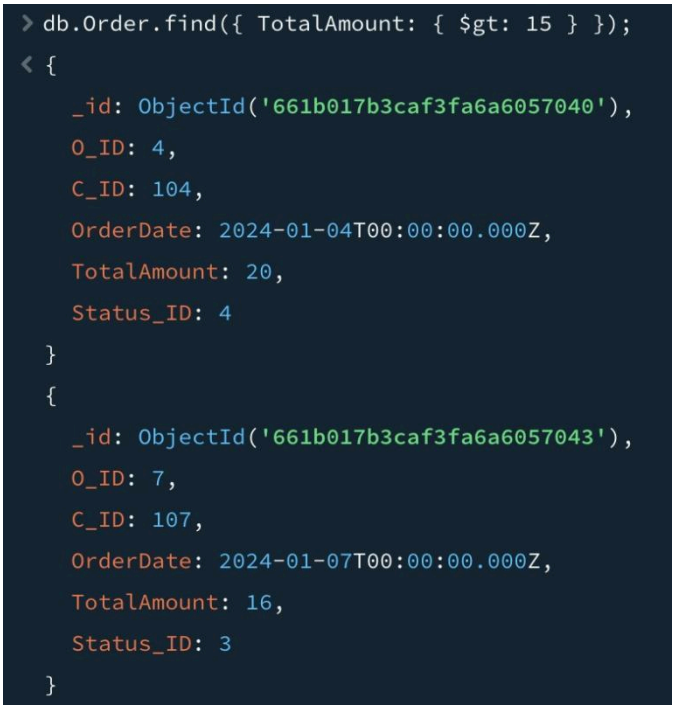
b. Find all customers who's name is John

db.Customer.find({ Name: "John" });

```
< {
    _id: ObjectId('661affd63caf3fa6a6056e99'),
    C_ID: 101,
    Name: 'John',
    Phone: '7890129876',
    Email: 'john@example.com',
    Address: '123 Main St'
  }
```

c. Find Orders with a Total Amount Greater than $15

db.Order.find({ TotalAmount: { $gt: 15 } });



QUERY 2: More Complex Query

Join Order and Customer collections to get order details along with the customer names:

```
db.Order.aggregate([
  {
   $lookup: {
    from: "Customer", // The collection to join with localField: "C_ID", // The field from the Order collection
    foreignField: "C_ID", // The matching field from the Customer collection as: "customer_info"
   }
  },
  {
   $unwind: "$customer_info" // Deconstructs the array field from the joined collection
  },
  {
   $project: { // Specifies the structure of the output documents
    _id: 1,
    Order_ID: 1,
    OrderDate: 1,
    TotalAmount: 1,
    Status_ID: 1,
   CustomerName: "$customer_info.Name", // Selects the customer name from the joined document
    CustomerEmail: "$customer_info.Email"
   }
```

}]);

Explanation: This complex query joins two collections to provide a detailed view of each order along with
the customer's name and email.

QUERY 3: Aggregate Query

To calculate the average total amount of orders:

```
db.Order.aggregate([
  {
    $group: {
      _id: null, // Grouping on a constant value to calculate the average across all documents
      AverageTotalAmount: { $avg: "$TotalAmount" } // The average operator to calculate the
average
    }
  }
]);
```

Explanation: This aggregate query calculates the average total amount for all the orders in the Order collection.

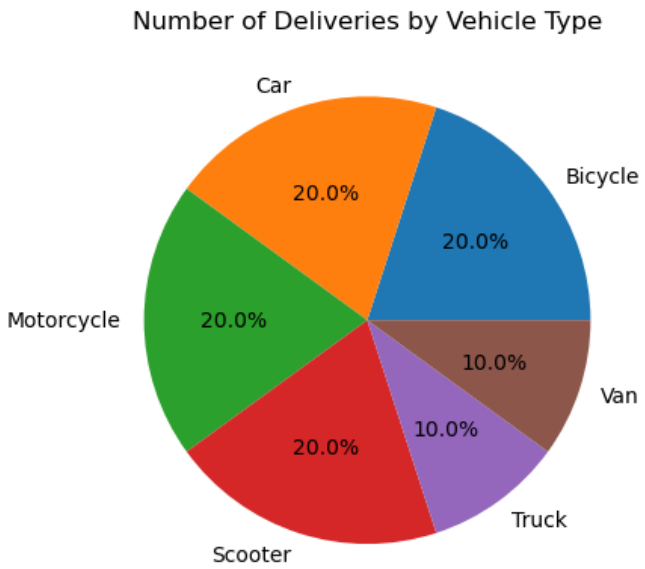
V. Database access via Python:

Our project has harnessed the power of SQL databases in conjunction with Python to perform comprehensive data extraction and analysis. Establishing a strong connection to a MySQL server, we've accessed a wealth of data that spans diverse aspects of our business operations, including chef specializations, delivery logistics, customer feedback, and sales performance.

Once the data was seamlessly imported into pandas dataframes, I turned my attention towards converting this complex information into clear, insightful visualizations. Employing tools such as Matplotlib and Seaborn, I've crafted a suite of visual narratives in the form of charts and graphs. These are more than mere illustrations; they unravel essential statistics and trends that raw data alone could not reveal.

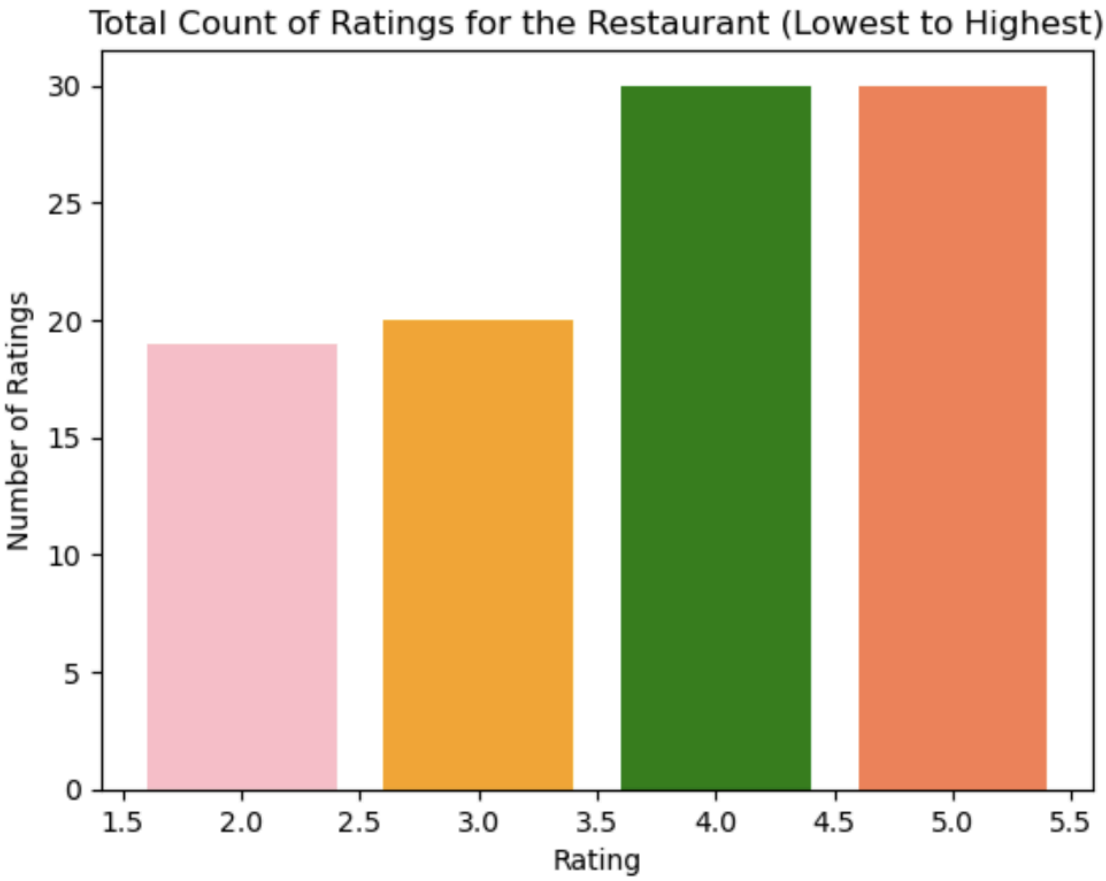
Each visualization has been thoughtfully tailored to explore distinct sectors of our business. Whether it's dissecting the array of delivery vehicles or interpreting the spectrum of customer ratings, these graphical representations enable us to identify our core competencies and areas ripe for development. Moreover, an in-depth look at sales data and promotional impact provides a foundation for strategic decision-making, rooted in a data-driven ethos.

Visualization 1: Number of Deliveries by Vehicle Type



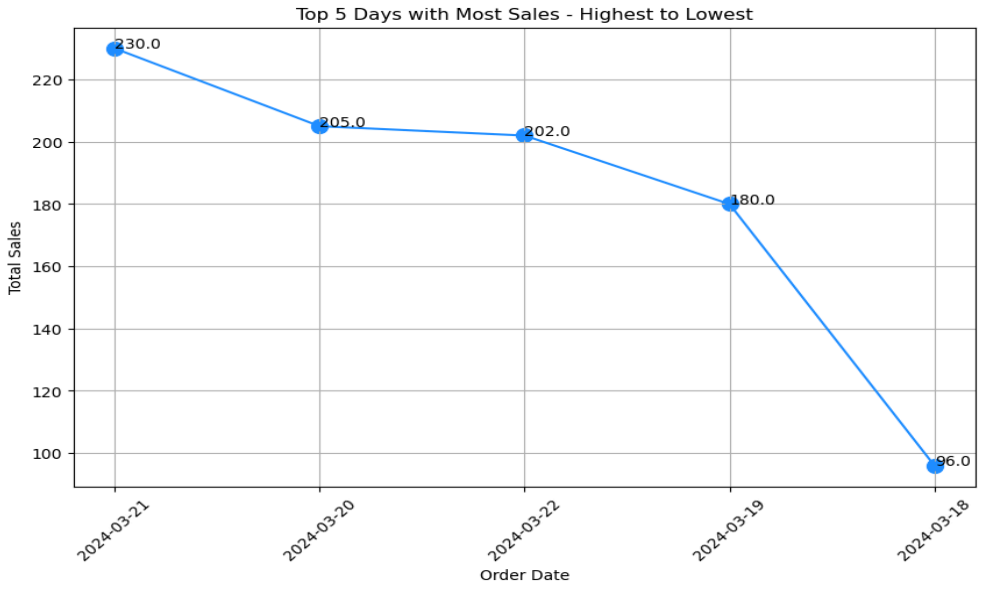
Insights:
The pie chart provides a distribution of deliveries by vehicle type, helping to identify which vehicles are used most frequently for deliveries.

Visualization 2: Total Count of Ratings for the Restaurant (Lowest to Highest)



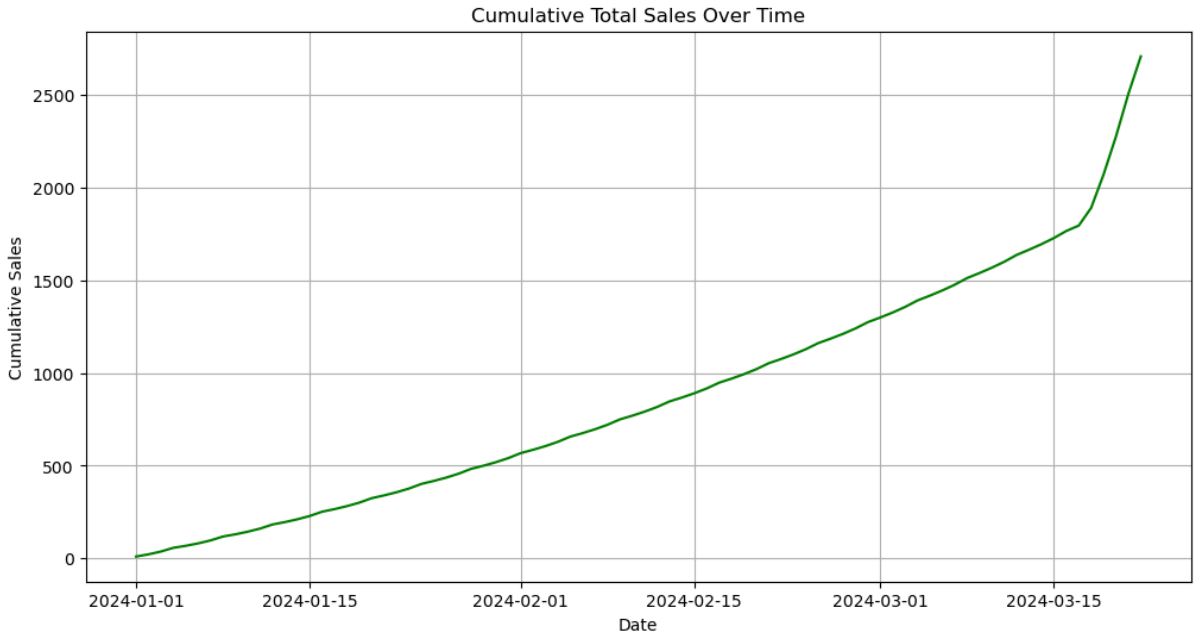
Insights:
The bar chart helps to visualize the distribution of customer ratings, highlighting areas where the restaurant performs well and areas that might need improvement.

Visualization 3: Top 5 Days with Most Sales



Insights:
This scatter plot shows the days with the highest sales, useful for understanding seasonal trends or the effects of promotions.

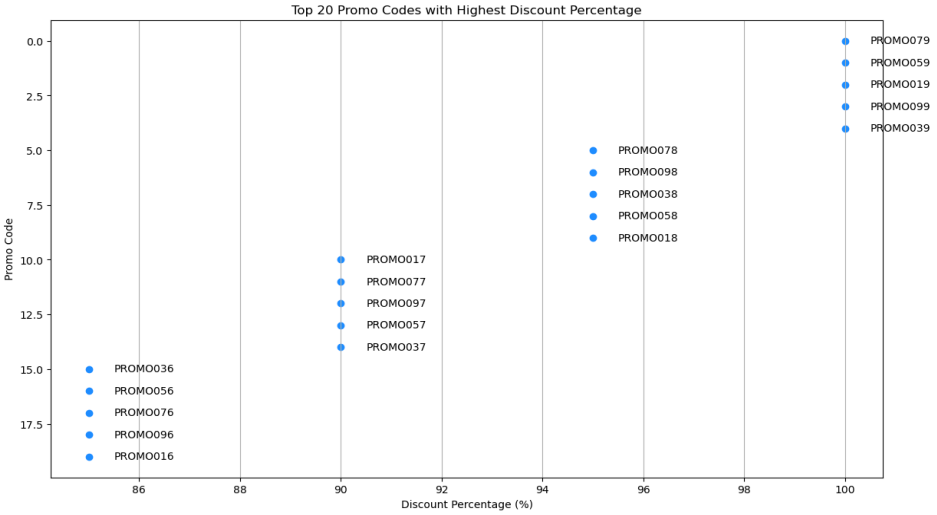
Visualization 4: Cumulative Total Sales Over Time



Insights:

The cumulative line plot of total sales over time indicates a consistent upward trajectory, signaling sustained growth in the business's sales performance over the period represented.

Visualization 5: Top 20 promo codes with the highest discount percentages



Insights:

The scatter plot visualizes the top 20 promo codes with the highest discount percentages offered by the business. The insight that can be drawn from this visualization is:
A small cluster of promo codes offers significantly higher discounts than the rest, which could be an indicator of aggressive promotional strategies aimed at boosting sales or clearing inventory.

VII. Summary and Recommendation:

Looking ahead, the project's next phase will focus on enhancing the capabilities of VirtuChef by migrating our operations to a cloud-based infrastructure. This strategic move is anticipated to bring transformative changes to the way we manage data and conduct business. A cloud-based system will provide us with the agility needed to scale operations, access data in real-time from any location, and employ advanced analytical tools for deeper insights into customer preferences and operational efficiency. The transition to the cloud also promises improved security measures for our valuable data and cost-effective solutions for managing our resources.

To ensure a smooth transition and capitalize on the benefits of cloud computing, we recommend a structured approach that encompasses careful planning, budgeting for necessary investments, and continuous monitoring post-migration. By prioritizing data security during this shift, we aim to establish a secure environment for our data assets. Additionally, we suggest setting up a centralized data warehouse in the cloud to consolidate data streams for better data management and analytics. Employing predictive analytics in our cloud
infrastructure will be instrumental in forecasting demand and optimizing our services.

The move to the cloud is not just a step towards improving current operations but is also a stride into the future of cloud kitchens, where scalability, efficiency, and security are paramount. With this in mind, we are committed to monitoring the performance of our cloud services diligently and making continuous improvements. Our goal is to ensure that VirtuChef remains at the forefront of innovation, setting new standards for operational excellence and customer satisfaction in the cloud kitchen industry.