**USE CASE STUDY REPORT**   
   
Group No: 8   
 Student Names:

Keshav Kumar Elankovan, Nidhi Jagadeesh, Simrat Kaur

**Executive Summary:**

The primary goal of this study was to design and implement an industry-ready relational database for handling the Baking-Cupcake industry used by any online retailer in the form of a cloud kitchen in the industry with complicated requirements based on the scenario defined for the specific products, handling huge data records, and that can be modified and replicated across other small scale food businesses. There is a lot of data duplication, discrepancy, and missing data, but this model can be reused by the properties by using a relational database.

This relational database significantly reduces data input process time, resulting in improved employee, order, and delivery management across the industry. The database also includes a central analytics platform with enormous potential for analytics on products, employees, and customer statistics, as well as a payment status check.

The database was designed with the attributes required by the business in mind, as well as only the fields required by the customer. The EER and UML diagrams were created, and then the conceptual model was mapped to a relational model with the necessary primary and foreign keys. This database was then fully implemented using the MySQL database, and tables were implemented on the MongoDB-NoSQL database to test its viability in a NoSQL environment.

The created database is a huge success, and by connecting it to Python, the analytics capabilities are enormous, as demonstrated in the study. These queries can be extremely beneficial in locating loyal customers, fast-moving products, and employees who perform exceptionally well. The next step would be to standardize this for other food industry specifications.

**I.Introduction**

Cupcake is a niche market in the baking industry, the statistics say that the industry turnover about $16 billion annually in the United States of America alone. The most inspiring and encouraging factor to getting into the cupcake industry is that you can start on a small scale from the comfort of your kitchen.

Boston cupcakes make celebrations taste better with premium baked fresh daily cupcakes since 2020. Boston cupcakes innovated an ever-growing, on-demand cupcake ATM, convenient for ordering online and Expanding. The cupcake business has a lot of fields like Delivery, Product Inventory, Orders, and Payment etc. It requires various attributes to handle both business and customer details . Resulting in large Scale data with duplication and time constraints.

Thus Relational database helps to manage and store the data. Help in Speed processes for Business to handle the inventory, employee, order, payment and delivery management, Which facilitates Customers to order any flavor, Toppings, and frosting of their choice. The Product will be checked in the database which has the inventory to prospect availability and displayed in the website. Furthermore, an order can be placed after the payment went through and a bill has been generated.

All these entities combined, provide The Boston Cupcakes with a simple database that addresses and solves management discrepancies and manage the delivery and employee management pipelines and process any problem arising in the cupcake business and also manage the orders and payment in an efficient Manner.

**II. Conceptual Data Modeling**

1. EER Diagram

Diagram

Description automatically generated

2. UML Diagram

Diagram, schematic

Description automatically generated

III. Mapping Conceptual Model to Relational Model   
   
Primary Key- Underlined Foreign Key- Italicized

Order (OrderId, Quantity)

Inventory (Available quantity, *ProductID*)

ProductID: Foreign Key from Product table, Not Null

Payment (Payment ID, Date, Status, *orderId*)

orderID : Foreign Key from Order table, Not Null

Bill ( Bill\_no , Amount)

Delivery ( Delivery no, Date, Time, *OrderID*)

Orderid: Foreign key-orders table, Not Null

Employee (Employee Id, name, phone no, Salary)

Customer (Customer ID, Name, Email, Phone no., Address, *Order Id*)

Order ID : Foreign Key from order table, Not Null

Standard (Description, *Product ID*)

ProductId: Foreign Key from payment table, Not Null

Customized (Frosting, additional topping, cake Flavor, *Product ID*)

ProductID: Foreign Key from Rent table, Not Null

baking (Employee Id, *Product Id*)

Orders-Product (orderid, *Product ID*)

delivery record (Employee Id, Delivery no)

IV. Implementation of Relation Model via MySQL and NoSQL   
   
MySQL Implementation:

Query 1: Top 3 employees with maximum count delivey

select Employee\_id, Employee\_name from employee

where Employee\_id IN

(select Employee\_id from delivery\_employee group by Employee\_id having count(\*) order by Employee\_id desc )

order by Employee\_id desc limit 3;

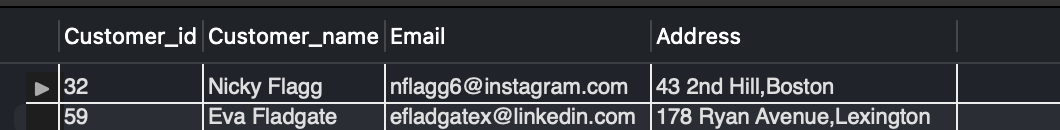
Graphical user interface, text, application

Description automatically generated

Query 2: Regular Customers who have multiple orders

Select \* from customer where Customer\_id

in (select customer\_id from orderplacement group by customer\_id having count(\*)>3);



Query 3: customer's list for payment failed order

select \* from customer where customer\_id in

(select o.customer\_id from orderplacement as o join payment as p

where o.order\_id=p.order\_id

and p.Status="False");

A picture containing text

Description automatically generated

Query 4: Revenue for the month of march,22

select sum(b.amount) as Revenue from bill as b join payment as p

where b.payment\_id=p.payment\_id and p.Status="True" and p.Date like "3/%/22";

Graphical user interface, text, application

Description automatically generated

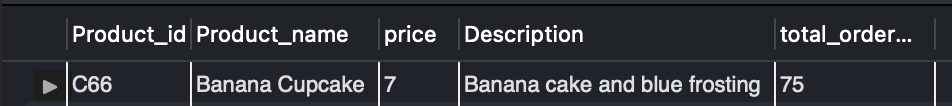
Query 5: Top product sold based on quantity

select p.Product\_id, p.Product\_name,price, p.Description,a.total\_ordered

from product p join

(select Product\_id, sum(quantity) as total\_ordered from orders group by Product\_id order by total\_ordered desc limit 1) a

where p.Product\_id=a.Product\_id;



Query 6: Products available in less quantity

select \* from product order by available\_quantity asc limit 3;

Table

Description automatically generated with medium confidence

NoSQL Implementation:

V. Database Access via Python

The visualizations provided are generated using Python. The connection between Mysql and Python was done using the pymysql library, where we collect the data and then using read\_sql\_query() function, convert the data into a pandas data frame. For plotting the data we use matplotlib for exploratory data analysis.

1) Number of times a customer ordered

Chart, histogram

Description automatically generated with medium confidence

2) Total quantity of a particular cupcake ordered product wise

Chart

Description automatically generated

3) The Amount paid by each customer

Text

Description automatically generated

Chart

Description automatically generated