ABSTRACT

The Grocery Management System (GMS) is an innovative solution designed to revolutionize the operations of grocery stores, supermarkets, and related establishments. In the modern digital era, managing inventory, sales, and customer interactions efficiently is crucial for success. GMS offers a comprehensive suite of features to streamline various aspects of grocery management, from inventory tracking to customer relationship management. It revolutionizes inventory management, streamlining stock tracking and replenishment processes in real time. Integrated with a seamless Point of Sale (POS) system, GMS ensures efficient and accurate transactions for customers, supporting various payment methods. Its Customer Relationship Management (CRM) feature enables personalized interactions, fostering loyalty and satisfaction through targeted marketing campaigns. Supplier management is enhanced with GMS, facilitating effective communication and order tracking. The system's analytics and reporting capabilities provide actionable insights into sales trends, inventory performance, and customer behavior, empowering data-driven decision-making. With mobile accessibility, GMS enables remote monitoring and management, enhancing operational flexibility. Overall, GMS equips grocery retailers with the tools needed to thrive in today's competitive market, optimizing efficiency and driving growth.

INDEX

CHAPTER I
INTRODUCTION

In the bustling world of grocery retail, efficiency is key to success. Introducing our Grocery Management System (GMS), a comprehensive solution designed to simplify the complexities of running a grocery store. With interconnected tables such as Customer, Product, Employee, Login, Supplier, Sales, Billing, and Category, GMS offers a holistic approach to store management. From tracking inventory and managing sales to ensuring secure access and facilitating seamless transactions, GMS streamlines every aspect of grocery store operations. With its user-friendly interface and powerful features, GMS empowers store owners to make informed decisions, enhance customer satisfaction, and drive business growth in today's competitive market landscape.

1.1 Software: MySQL

The back-end database used in this project is MySQL

It is a language used to interrogate and process data in a relational database. Originally developed by IBM for its mainframes, SQL commands can be used to interactively work with a database or can be embedded within a script or programming language as an interface to a database. Programming extensions to SQL have turned it into a full-blown database programming language, and all major database management systems (DBMSs) support it.

ANSI standardized SQL.

But most DBMSs have some proprietary enhancement, which if used, makes SQL non-standard. Moving an application from one SQL database to another sometimes requires tweaking, an age-old problem in this business!

1.2 Advantages of MySQL

1. SQL Queries can be used to retrieve large amounts of records from a database quickly.
2. SQL is used to view the data without storing the data in the object
3. SQL joins two or more tables and shows it as one object to the user
4. SQL databases use long-established standards, which are being adopted by ANSI & ISO. Non-SQL databases do not adhere to any clear standard.

CHAPTER II

PROJECT FEATURES AND OBJECTIVES

2.1 MAIN FEATURES AND FUNCTIONALITY

1. **Inventory Management:** Efficiently track and manage inventory levels, including stock quantities, product details, and categories. Implement automated alerts for low stock levels and optimize reorder processes.
2. **Point of Sale (POS) System:** Streamline checkout processes with a user-friendly POS system. Support multiple payment methods, including cash, credit/debit cards, and mobile payments. Generate digital receipts and manage returns seamlessly.
3. **Customer Relationship Management (CRM):** Capture and store customer information, including purchase history, preferences, and contact details. Implement loyalty programs, targeted promotions, and personalized marketing campaigns to enhance customer engagement and retention.
4. **Supplier Management:** Maintain a database of suppliers/vendors, including contact information, pricing details, and order history. Streamline communication and collaboration with suppliers, track orders, and manage delivery schedules effectively.
5. **Sales and Reporting:** Record and analyze sales transactions in real time. Generate comprehensive reports and analytics to gain insights into sales performance, product trends, and customer behavior. Use data-driven insights to make informed business decisions and drive growth.
6. **Billing and Accounting:** Manage billing processes efficiently, including invoicing, billing cycles, and payment tracking. Integrate with accounting software to streamline financial operations and ensure accurate reporting.
7. **Employee Management:** Maintain employee records, including contact information, roles, and schedules. Manage permissions and access levels within the system to ensure security and compliance.

8. **Category Management:** Organize products into logical categories and subcategories to enhance navigation and improve the shopping experience for customers. Easily update and modify category structures as needed.
9. **Mobile Accessibility:** Access the system remotely via mobile devices to monitor operations, track inventory, and manage tasks on the go. Stay connected and responsive to store needs at all times.
10. **Security and Access Control:** Implement robust security measures, including user authentication, role-based access control, and data encryption, to safeguard sensitive information and ensure compliance with regulatory requirements.

2.2 OBJECTIVES

1. **Efficient Inventory Management:** Streamline the tracking and management of inventory levels, reducing stockouts and minimizing wastage through optimized replenishment processes.
2. **Enhanced Customer Experience:** Provide personalized service and targeted promotions based on customer preferences and purchase history, fostering loyalty and satisfaction.
3. **Improved Operational Efficiency:** Automate routine tasks such as inventory tracking, order management, and billing processes to reduce manual workload and increase productivity.
4. **Effective Supplier Collaboration:** Facilitate seamless communication and collaboration with suppliers to ensure timely deliveries, negotiate favorable terms, and maintain high product quality.
5. **Data-driven decision-making:** Generate actionable insights through comprehensive reporting and analytics, enabling informed business decisions to drive growth and profitability.
6. **Streamlined Point of Sale (POS) Operations:** Simplify checkout processes for customers and staff with an intuitive POS system, supporting multiple payment methods and ensuring accuracy in transactions.

7. **Secure Data Management:** Implement robust security measures to protect sensitive information, including customer data, financial records, and proprietary business information, from unauthorized access or breaches.
8. **Optimized Employee Management:** Effectively manage employee schedules, roles, and permissions within the system, ensuring smooth operations and adherence to labor regulations.
9. **Flexible and Scalable Solution:** Offer a scalable platform that can adapt to the evolving needs of grocery stores, accommodating growth, technological advancements, and changing market dynamics.
10. **Enhanced Mobile Accessibility:** Enable remote access to key functionalities via mobile devices, empowering store managers to monitor operations, track inventory, and address issues from anywhere, at any time.

2.3 IDENTIFICATION OF PROJECT MODULES

1. Inventory Management Module
2. Point of Sale (POS) Module
3. Customer Management Module
4. Supplier Management Module
5. Reporting and Analytics Module

2.4 MODULE DESCRIPTION

2.4.1 Module1

Inventory Management Module: This module is the backbone of the system, allowing users to track and manage inventory levels efficiently. It includes features such as stock tracking, product categorization, automated reordering, and inventory optimization algorithms to ensure optimal stock levels and minimize wastage.

2.4.2 Module 2

Point of Sale (POS) Module: The POS module is responsible for handling all aspects of the checkout process. It includes features such as scanning items, calculating prices, processing various payment methods (cash, credit/debit cards, mobile payments), generating receipts, and managing returns. The POS module ensures smooth and efficient transactions for both customers and staff.

2.4.3 Module 3

Customer Management Module: This module focuses on managing customer relationships and enhancing the shopping experience. It allows users to capture and store customer information such as contact details, purchase history, preferences, and feedback. With this data, the system can personalize marketing efforts, implement loyalty programs, and provide targeted promotions to improve customer satisfaction and retention.

2.4.4 Module 4

Supplier Management Module: The Supplier Management module centralizes information related to suppliers and vendor relationships. It includes features such as maintaining supplier contact details, managing pricing agreements, tracking order history, and monitoring delivery schedules. This module streamlines communication with suppliers, ensures timely deliveries, and helps maintain high product quality.

2.4.5 Module 5

Reporting and Analytics Module: The Reporting and Analytics module provides users with valuable insights into business performance and trends. It includes features such as generating customizable reports, analyzing sales data, monitoring inventory turnover rates, and identifying opportunities for growth and optimization. By leveraging data-driven insights, users can make informed decisions to drive profitability and strategic planning.

CHAPTER III
BACK-END DESIGN AND CONNECTIVITY

3.1 BACK-END DESIGN

The backend design of a Grocery Management System is essential for managing data effectively and implementing the business logic that governs system operations. It ensures efficient storage, retrieval, and manipulation of data related to inventory, sales, customers, and suppliers. A well-designed backend architecture enables the system to handle large volumes of data, execute complex algorithms for tasks such as inventory management and pricing calculations, and maintain reliability and scalability as the system grows.

3.1.1 Conceptual Database Design (ER-Diagram)

Fig 3.1.1 illustrates the comprehensive ER Diagram depicting various entities, attributes, and relationships between the entities

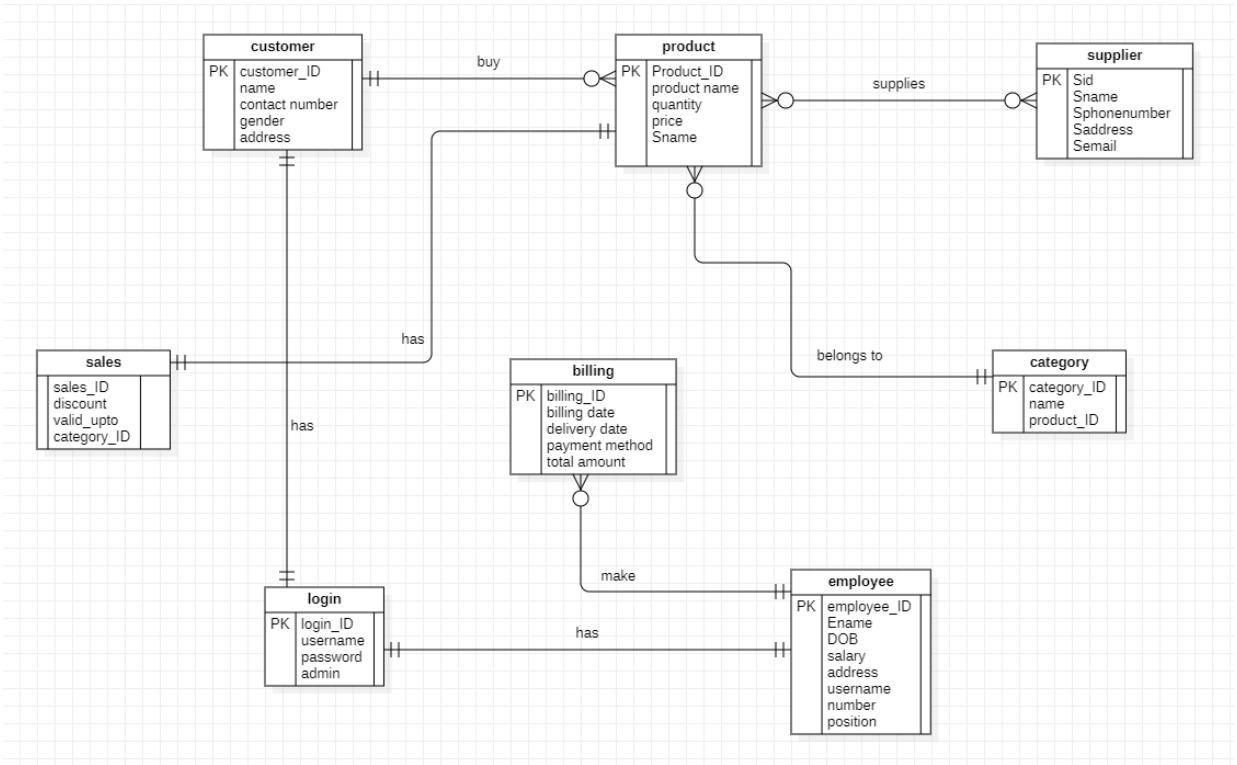

Fig 3.1 ER Diagram

3.1.2 Logical Database Design (ER Mapping)

SCHEMA DIAGRAM

Fig 3.1.2 illustrates the workflow of the operation performed in the Grocery Management System.

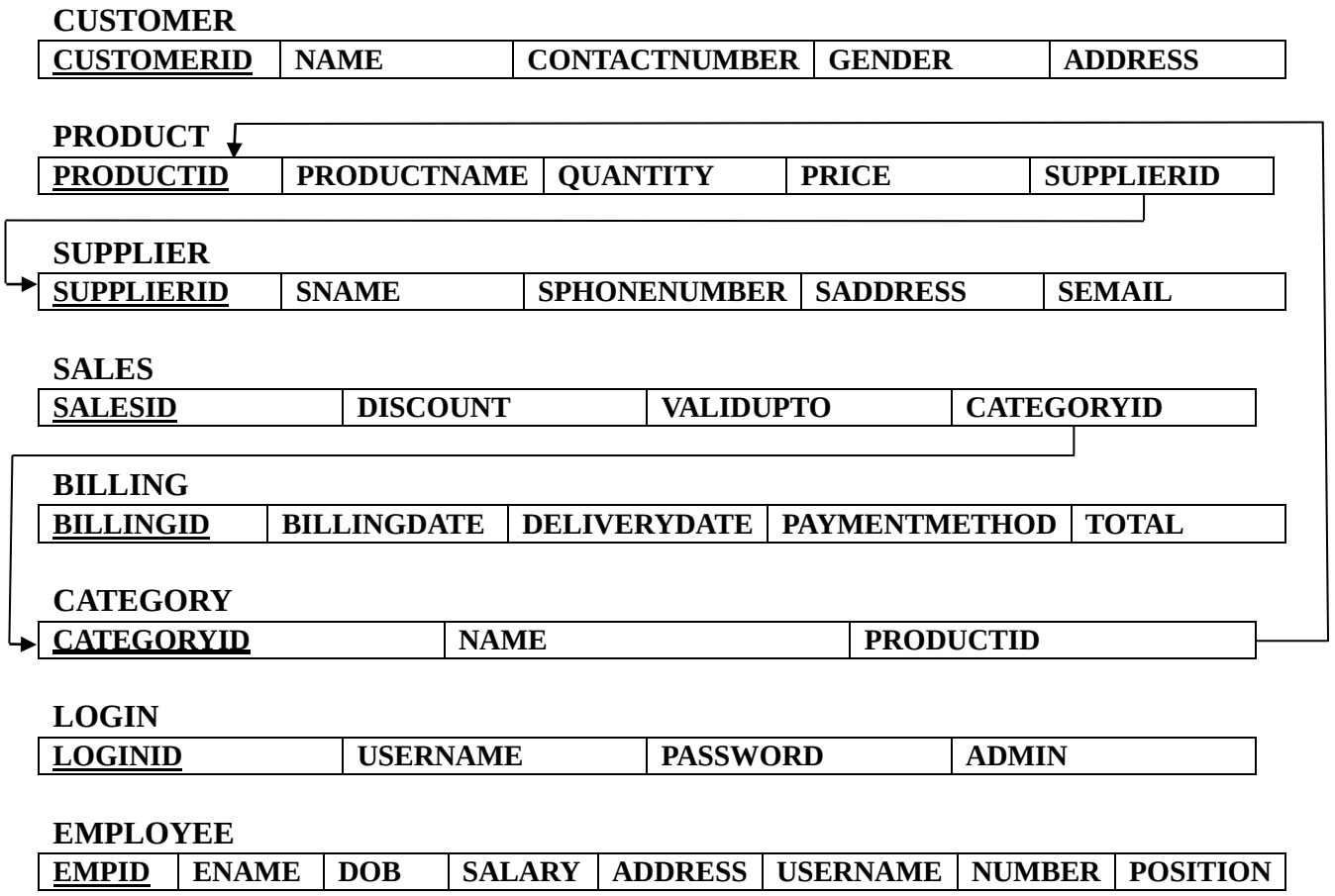

Fig 3.2 Schema diagram

- The entities are represented as tables.
- The tables contain the attributes.
- The attributes which are underlined are referred as primary keys.

3.2 FRONT-END DESIGN

3.2.1 Front-end web development details

- **HTML** provides the basic structure of sites, which is enhanced and modified by other technologies like CSS and JavaScript.
- **CSS** is used to control presentation, formatting, and layout.
- **JavaScript** is used to control the behavior of different elements.

HTML

HTML is at the core of every web page, regardless of the complexity of a site or a number of technologies involved. It's an essential skill for any web professional. It's the starting point for anyone learning how to create content for the web. And, luckily for us, it's surprisingly easy to learn.

CSS

CSS stands for Cascading Style Sheets. This programming language dictates how the HTML elements of a website should appear on the front end of the page.

JavaScript

JavaScript is a more complicated language than HTML or CSS, and it wasn't released in beta form until 1995. Nowadays, JavaScript is supported by all modern web browsers and is used on almost every site on the web for more powerful and complex functionality.

CHAPTER IV

MODULE IMPLEMENTATION

4.1 CONSTRUCTION OF RELATIONAL TABLE FROM THE ER DIAGRAM

Fig 4.1 shows various tables which are developed in MySQL workbench

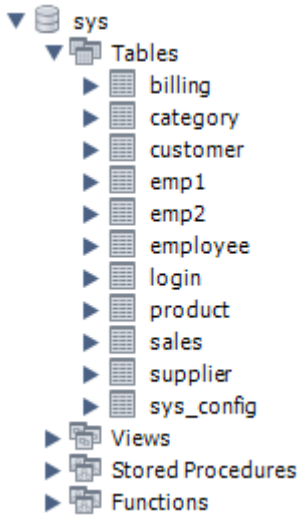

Fig 4.1 Tables list

4.1.1 DDL, DML, DCL, TCL of Grocery Management System

DDL:

CUSTOMER:

Create table customer (customerId int primary key, customerName varchar(45), customernumber int, gender varchar(45), address varchar(45));

ALTER TABLE Customer MODIFY COLUMN gender CHAR(100);
ALTER TABLE Customer ADD COLUMN email VARCHAR(100);
ALTER TABLE Customer DROP COLUMN customer_number;
ALTER TABLE Customer CHANGE COLUMN customername full_name VARCHAR(50);

ALTER TABLE Customer ADD CONSTRAINT chk_gender CHECK (gender IN ('Male', 'Female', 'Other'));

DROP TABLE Customer;

PRODUCT:

CREATE TABLE Product (Product_ID INT PRIMARY KEY AUTO_INCREMENT, Product_name VARCHAR(100) NOT NULL, quantity INT NOT NULL, price DECIMAL(10, 2) NOT NULL, supplier_id INT, FOREIGN KEY (supplier_id) REFERENCES Supplier(supplier_id));

ALTER TABLE Product ADD COLUMN expiration_date DATE;
ALTER TABLE Product MODIFY COLUMN price DECIMAL(12, 2);
ALTER TABLE Product CHANGE COLUMN Product_name product_name VARCHAR(150);

DROP TABLE IF EXISTS Product;

SUPPLIER:

CREATE TABLE Supplier (Supplier_id INT PRIMARY KEY AUTO_INCREMENT, Sname VARCHAR(100) NOT NULL, SPhonenumber VARCHAR(20) NOT NULL, Semail VARCHAR(100), Saddress VARCHAR(200) NOT NULL);

ALTER TABLE Supplier ADD COLUMN contact_person VARCHAR(100);
ALTER TABLE Supplier DROP COLUMN Semail;
ALTER TABLE Supplier MODIFY COLUMN SPhonenumber VARCHAR(15);
ALTER TABLE Supplier CHANGE COLUMN Sname supplier_name VARCHAR(150);

DROP TABLE IF EXISTS Supplier;

SALES:

CREATE TABLE SALES (sales_ID INT PRIMARY KEY AUTO_INCREMENT, discount DECIMAL(5, 2) NOT NULL, valid_upto DATE,category_ID INT, FOREIGN KEY (category_ID) REFERENCES Category(category_ID));

ALTER TABLE SALES ADD COLUMN sale_date DATE;
ALTER TABLE SALES DROP COLUMN valid_upto;
ALTER TABLE SALES MODIFY COLUMN discount DECIMAL(6, 2);
ALTER TABLE SALES CHANGE COLUMN sales_ID sale_id INT;

DROP TABLE IF EXISTS SALES;

BILLING:

CREATE TABLE Billing (billing_ID INT PRIMARY KEY AUTO_INCREMENT, billing_date DATE NOT NULL, delivery_date DATE, payment_method VARCHAR(50) NOT NULL, total_amount DECIMAL(10, 2) NOT NULL);

ALTER TABLE Billing ADD COLUMN customer_id INT;
ALTER TABLE Billing DROP COLUMN delivery_date;
ALTER TABLE Billing CHANGE COLUMN total_amount bill_amount DECIMAL(12, 2);

DROP TABLE IF EXISTS Billing;

CATEGORY:

CREATE TABLE Category (category_id INT PRIMARY KEY AUTO_INCREMENT, name VARCHAR(100) NOT NULL, product_id INT, FOREIGN KEY (product_id) REFERENCES Product(Product_ID));

ALTER TABLE Category ADD COLUMN description VARCHAR(255);
ALTER TABLE Category DROP COLUMN product_id;
ALTER TABLE Category MODIFY COLUMN name VARCHAR(150);

DROP TABLE IF EXISTS Category;

LOGIN:

CREATE TABLE Login (login_id INT PRIMARY KEY AUTO_INCREMENT, username VARCHAR(50) NOT NULL UNIQUE, password VARCHAR(50) NOT NULL, admin BOOLEAN NOT NULL);

ALTER TABLE Login ADD COLUMN email VARCHAR(100);
ALTER TABLE Login DROP COLUMN admin;
ALTER TABLE Login MODIFY COLUMN password VARCHAR(255);

DROP TABLE IF EXISTS Login;

EMPLOYEE:

CREATE TABLE Employee (employee_id INT PRIMARY KEY AUTO_INCREMENT, ename VARCHAR(100) NOT NULL, dob DATE, salary DECIMAL(10, 2) NOT NULL, address VARCHAR(200) NOT NULL, username VARCHAR(50) UNIQUE, phone_number INT NOT NULL, position VARCHAR(100), email VARCHAR(100));

ALTER TABLE Employee ADD COLUMN hire_date DATE;
ALTER TABLE Employee DROP COLUMN email;
ALTER TABLE Employee CHANGE COLUMN position job_title VARCHAR(100);

DROP TABLE IF EXISTS Employee;

EMP1:

CREATE TABLE Emp1 (eid INT PRIMARY KEY AUTO_INCREMENT, ename VARCHAR(100) NOT NULL, gender ENUM('Male', 'Female', 'Other') NOT NULL, salary DECIMAL(10, 2), department VARCHAR(100));

EMP2:

CREATE TABLE Emp2 (eid INT PRIMARY KEY AUTO_INCREMENT, ename VARCHAR(100) NOT NULL, gender ENUM('Male', 'Female', 'Other') NOT NULL, salary DECIMAL(10, 2), department VARCHAR(100));

DML, DCL, TCL:

```
SELECT * FROM sys.login;

INSERT INTO Login VALUES (1, 'user1', 'yuwtgdqy', 0);
INSERT INTO Login VALUES (2, 'admin1', 'yagyrquiy98', 1);
INSERT INTO Login VALUES (3, 'custome2', 'dyuqu3y', 0);

DELETE FROM Login WHERE login_id = 5;
UPDATE Login SET password = 'new_password' WHERE username = 'user1';

GRANT SELECT ON Login TO user1;
REVOKE SELECT ON Login FROM user1;

TRUNCATE table Login;
Commit;
```

4.1.2 In- Built functions of Grocery Management System

SALES:

```
select * from sys.sales where discount between 10 and 20;
select sales_ID from sys.sales where category_ID between 2 and 6;
select * from sys.sales where sales_ID=10;
select sqrt(discount) from sys.sales;
select max(discount) as max_discount, min(discount) as min_discount from sys.sales;
select sum(discount) from sys.sales;
SELECT COUNT(*) AS Totalsales FROM sys.sales;
```

CATEGORY:

```
SELECT * FROM sys.category;
select name from sys.category where category_ID between 2 and 6;
select * from sys.category where name like 'b%';
select * from sys.category where name not like 'b%';
select sum(product_ID) from sys.category;
SELECT COUNT(*) AS Totalcategories FROM sys.category;
```

PRODUCT:

```
SELECT * FROM sys.product;
select * from sys.product where Product_name like 'b%';
select * from sys.product where Product_name not like 'b%';
select sum(price) from sys.product;
select sqrt(quantity) from sys.product;
select max(price) as max_price, min(price) as min_price from sys.product;
SELECT COUNT(*) AS Totalproducts FROM sys.product;
select Product_name,min(price) from sys.product group by Product_name;
```

EMPLOYEE:

```
SELECT * FROM sys.employee;
select sqrt(salary) from sys.employee;
select * from sys.employee where salary between 50000 and 70000;
select max(salary) as max_salary, min(salary) as min_salary from sys.employee;
select sum(salary) from sys.employee;
select ename,min(salary) from sys.employee group by ename;
select * from sys.employee where ename like 'a%';
select * from sys.employee where ename not like 'a%';
SELECT COUNT(*) AS TotalRecords FROM sys.employee;
```

SUPPLIER:

```
SELECT * FROM sys.supplier;
SELECT COUNT(*) AS Total FROM sys.supplier;
select * from sys.supplier where Sname like 'm%';
select * from sys.supplier where Sname not like 'm%';
select * from sys.supplier where Supplier_id between 19 and 24;
```

BILLING:

```
SELECT * FROM sys.billing;
SELECT COUNT(*) AS Totalbillings FROM sys.billing;
select * from sys.billing where payment_method like 'u%';
select * from sys.billing where payment_method='Cash';
select sum(total_amount) from sys.billing;
select max(total_amount) as max, min(total_amount) as min from sys.billing;
select sqrt(total_amount) from sys.billing;
select * from sys.billing where total_amount between 50 and 200;
```

LOGIN:

SELECT * FROM sys.login;

SELECT COUNT(*) AS TotalRecords FROM sys.login;

select * from sys.login where password like 'a%';

select * from sys.login where password not like 'a%';

select * from sys.login where login_ID between 5 and 7;

CUSTOMER:

SELECT * FROM sys.customer;

SELECT COUNT(*) AS Totalcustomers FROM sys.customer;

select * from sys.customer where customername like 'c%';

select * from sys.customer where customername not like 'c%';

select * from sys.customer where customerid between 105 and 107;

4.1.3 Nested Queries of Grocery Management System

SALES:

select valid_upto from sys.sales where sales_ID=(select max(sales_ID) from sys.sales);

select * from sys.sales where discount > (select avg(discount) from sys.sales);

select * from sys.sales where discount > (select avg(discount) from sys.sales) order by discount desc;

select * from sys.sales where sales_ID in(select sales_ID from sys.sales where discount>=20);

UPDATE sys.sales SET discount = discount * 0.25 WHERE sales_ID IN (SELECT sales_ID FROM (SELECT sales_ID FROM sys.sales WHERE sales_ID >= 18) AS subquery);

CATEGORY:

select name from sys.category where category_ID=(select max(category_ID) from sys.category);

select * from sys.category where product_ID > (select avg(product_ID) from sys.category);

select * from sys.category where product_ID > (select avg(product_ID) from sys.category) order by product_ID desc;

select category_ID,name from sys.category where category_ID in(select category_ID from sys.category where product_ID>=15);

PRODUCT:

select Product_name from sys.product where price=(select max(price) from sys.product);

select * from sys.product where price > (select avg(price) from sys.product);

select * from sys.product where price > (select avg(price) from sys.product) order by price desc;

select Product_name, quantity, price from sys.product where Product_ID in(select product_ID from sys.product where product_ID>=15);

EMPLOYEE:

select ename, username, position from sys.employee where salary=(select max(salary) from sys.employee);

select * from sys.employee where salary > (select avg(salary) from sys.employee);

select * from sys.employee where salary > (select avg(salary) from sys.employee) order by salary desc;

select ename, salary, phone_number, position from sys.employee where employee_ID in(select employee_ID from sys.employee where employee_ID>=5);

select * from sys.employee where address in(select address from sys.employee where address='potheri');

SUPPLIER:

select Sname, SPhonenumber, Semail from sys.supplier where Supplier_id=(select max(Supplier_id) from sys.supplier);

select * from sys.supplier where Supplier_id > (select avg(Supplier_id) from sys.supplier);

select * from sys.supplier where Supplier_id > (select avg(Supplier_id) from sys.supplier) order by Supplier_id desc;

select Sname, SPhonenumber, Saddress from sys.supplier where Supplier_id in(select Supplier_id from sys.supplier where Supplier_id>=22);

select Supplier_id, SPhonenumber, Saddress from sys.supplier where Sname in(select Sname from sys.supplier where Sname like 'm%');

BILLING:

select billing_date, delivery_date, billing_ID from sys.billing where billing_ID=(select max(billing_ID) from sys.billing);

select * from sys.billing where billing_ID > (select avg(billing_ID) from sys.billing);

select * from sys.billing where billing_ID > (select avg(billing_ID) from sys.billing) order by billing_ID desc;

select delivery_date, billing_ID, total_amount from sys.billing where billing_ID in(select billing_ID from sys.billing where billing_ID>=4);

select * from sys.billing where payment_method in(select payment_method from sys.billing where payment_method='Cash');

LOGIN:

select username,password from sys.login where login_ID=(select max(login_ID) from sys.login);

select * from sys.login where login_ID > (select avg(login_ID) from sys.login);

select * from sys.login where login_ID > (select avg(login_ID) from sys.login) order by login_ID desc;

select username,password, admin from sys.login where login_ID in(select login_ID from sys.login where login_ID between 5 and 8);

select * from sys.login where username in(select username from sys.login where username='admin3');

CUSTOMER:

select customername, customer_number, gender from sys.customer where customerid=(select max(customerid) from sys.customer);

select * from sys.customer where customerid > (select avg(customerid) from sys.customer) order by customerid desc;

select customerid, customername, address from sys.customer where customerid in(select customerid from sys.customer where customerid>=5);

select * from sys.customer where address in(select address from sys.customer where address='potheri');

select * from sys.customer where customername in(select customername from sys.customer where customername like 'c%');

4.1.4 Set Operators & Views of Grocery Management System

ON EMP1 AND EMP2:

select * from sys.emp1 where salary>=30000 union select * from sys.emp2 where salary>=30000;

select * from sys.emp1 where ename like 'r%' union all select * from sys.emp2 where ename like 'r%';

SELECT * FROM sys.emp1 WHERE salary <= 30000 AND (salary) NOT IN (SELECT salary FROM sys.emp2 WHERE salary <= 30000) ORDER BY salary ASC;

VIEWS:

```
create view sorted_by_name as select eid, ename, gender, salary from sys.emp2 order by ename asc;
select * from sorted_by_name;
select count(*) as count from sorted_by_name;
drop view sorted_by_name;
```

4.1.5 PL/SQL Procedures and Functions of Grocery Management System

PROCEDURES:

EMPLOYEE:

```
use sys;
select * from employee;
call new_procedure(@records,30000);
select @records as total_records;
```

BILLING:

```
DELIMITER //

CREATE PROCEDURE sys.CalculateTotalAmount (
    OUT p_total_amount DECIMAL(10, 2)
)
BEGIN
    DECLARE total DECIMAL(10, 2);

    SELECT SUM(total_amount) INTO total
    FROM sys.billing;
```

```
    SET p_total_amount = total;
END //

DELIMITER ;



use sys;
SELECT * FROM sys.billing;
CALL sys.CalculateTotalAmount(@total_amount);
SELECT @total_amount AS TotalAmount;
```

FUNCTIONS:

CUSTOMER:

```
DELIMITER //

CREATE FUNCTION sys.CountGender(
    p_gender VARCHAR(10)  -- 'Male' or 'Female'
)
RETURNS INT
READS SQL DATA
BEGIN
    DECLARE count_gender INT;

    SELECT COUNT(*) INTO count_gender
    FROM sys.customer
    WHERE gender = p_gender;

    RETURN count_gender;
END //
```

```
DELIMITER ;


SELECT sys.CountGender('Male');
SELECT sys.CountGender('Female');
```

LOGIN:

```
DELIMITER //

CREATE FUNCTION sys.PrintAdmins()
RETURNS VARCHAR(255)
READS SQL DATA
BEGIN
   DECLARE admin_list VARCHAR(255);

   SELECT GROUP_CONCAT(username) INTO admin_list
   FROM sys.login
   WHERE admin = 1;

   RETURN admin_list;
END //

DELIMITER ;


SELECT sys.PrintAdmins();
```

4.1.6 PL/SQL Cursors and Exceptional Handling of Grocery Management System

CURSORS:

EMPLOYEE:

```
DELIMITER $$

CREATE PROCEDURE sys.proc_emp()
BEGIN
   DECLARE v_ename VARCHAR(100);
   DECLARE v_salary INT;
   DECLARE v_result VARCHAR(1000); -- Variable to store concatenated result

   DECLARE v_finished INTEGER DEFAULT 0;

   DECLARE c1 CURSOR FOR SELECT ename, salary FROM sys.employee;

   DECLARE CONTINUE HANDLER FOR NOT FOUND SET v_finished = 1;
   OPEN c1;

   get_emp: LOOP
      FETCH c1 INTO v_ename, v_salary;
      IF v_finished = 1 THEN
         LEAVE get_emp;
      END IF;
```

```
        SET v_result = CONCAT_WS(', ', v_result, CONCAT(v_ename, ' ', v_salary)); -- Concatenate values

    END LOOP get_emp;
    CLOSE c1;
    SELECT v_result; -- Display concatenated result
END $$

DELIMITER ;

CALL sys.proc_emp();
```

EXCEPTION HANDLING:

EMP1:

```
DELIMITER $$

CREATE PROCEDURE sys.InsertEmployee(
    IN p_eid INT,
    IN p_ename VARCHAR(100),
    IN p_gender CHAR(1),
    IN p_salary DECIMAL(10, 2),
    IN p_department VARCHAR(100)
)
BEGIN
    DECLARE exit handler for sqlexception
    BEGIN
        ROLLBACK;
        SELECT 'Error: An error occurred during insertion';
    END;
```

```
    DECLARE exit handler for sqlwarning
    BEGIN
        ROLLBACK;
        SELECT 'Warning: A warning occurred during insertion';
    END;

    START TRANSACTION;

    INSERT INTO sys.emp1 (eid, ename, gender, salary, department)
    VALUES (p_eid, p_ename, p_gender, p_salary, p_department);

    COMMIT;
    SELECT 'Employee inserted successfully';
END $$
DELIMITER ;


CALL sys.InsertEmployee(10, 'John Doe', 'M', 50000.00, 'finance');
select * from sys.emp1;
```

LOGIN:

```
DELIMITER $$

CREATE PROCEDURE sys.ValidateLogin(
    IN p_username VARCHAR(100),
    IN p_password VARCHAR(100)
)
BEGIN
    DECLARE v_admin INT;
```

```
    DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
        SELECT 'Error: An error occurred while validating login';
    END;

    DECLARE EXIT HANDLER FOR SQLWARNING
    BEGIN
        SELECT 'Warning: A warning occurred while validating login';
    END;
    START TRANSACTION;

    SELECT admin INTO v_admin
    FROM sys.login
    WHERE username = p_username OR password = p_password;

    IF v_admin IS NOT NULL THEN
        SELECT 'Login successful. Admin status: ' || v_admin;
    ELSE
        SELECT 'Invalid username or password';
    END IF;
    COMMIT;
END $$
DELIMITER ;


CALL sys.ValidateLogin('user1', 'yuwtgdqy');
```

4.2 OUTPUT

4.1.1-DDL, DML, DCL, TCL

Result Grid | Filter Rows: | Edit:

customerid	customername	customer_number	gender	address
101	alex	8618	male	Potheri
102	navya	9864	female	Potheri
103	durga	6981	female	Guduvancheri
104	alex	7492	male	Avadi
105	chandu	1984	female	avadi
106	reena	9478	female	potheri
107	arush	2349	male	tambaram
108	charvik	1203	male	TNagar
109	varsha	5077	female	Puducherry
110	jayapal	9477	male	Potheri
NULL	NULL	NULL	NULL	NULL

Product_ID	product_name	quantity	price	Supplier_id	expiration_date
12	apples	100	2.99	19	NULL
13	milk	50	1.79	20	NULL
14	bread	75	2.49	21	NULL
15	eggs	120	3.99	22	NULL
16	chicken	30	5.99	23	NULL
17	rice	40	8.49	24	NULL
23	bananas	110	1.95	25	NULL

Supplier_id	Sname	SPhonenumber	Semail	Saddress
19	Fresh Fruits Inc	92557	a@gmail.com	Banglore
20	Dairy Delights Co	24235	b@gmail.com	karnataka
21	Bakers Best	64788	c@gmail.com	Guntur
22	Farm Fresh Eggs	21434	d@gmail.com	Hyderabad
23	Meat Master	56789	e@gmail.com	Vijayawada
24	Grain Growers	12346	f@gmail.com	Guntur
25	Fresh Veggies Inc	42465	g@gmail.com	Guntur
26	Fresh Fruits	88769	h@gmail.com	Banglore
27	Baker Wow	98364	i@gmail.com	Hyderabad
28	Mushroom mama	9684	j@gmail.com	Mumbai

discount	valid_upto	category_ID	sales_ID
10	2024-04-01 00:00:00	1	7
15	2024-03-15 00:00:00	2	8
20	2024-04-30 00:00:00	3	9
8	2024-03-25 00:00:00	4	10
12	2024-04-10 00:00:00	5	11
25	2024-03-20 00:00:00	6	12
18	2024-04-15 00:00:00	7	13

billing_date	payment_method	total_amount	billing_ID	customer_id
2024-02-05 00:00:00	Credit Card	150	1	NULL
2024-02-12 00:00:00	Cash	92	2	NULL
2024-02-20 00:00:00	Online transfer	210	3	NULL
2024-02-28 00:00:00	Debit Card	75	4	NULL
2024-03-05 00:00:00	Cash	100	5	NULL
2024-03-15 00:00:00	Debit Card	125	6	NULL
2024-03-24 00:00:00	UPI	250	7	NULL
2024-04-02 00:00:00	UPI	450	8	NULL
2024-04-10 00:00:00	Credit Card	700	9	NULL
2024-04-15 00:00:00	Cash	1000	10	NULL

category_ID	name	product_ID
1	Fruits	12
2	Dairy	13
3	Bakery	14
4	Eggs	15
5	Meat	16
6	Grains	17
7	Fruits	23

login_ID	username	password	admin
1	user1	yuwtgdqy	0
2	admin1	yagyrquiy98	1
3	custome2	dyuqu3y	0
4	admin2	qjhgduyq	1
5	admin3	dvhafd3	1
6	user3	ayuxdqw6	0
7	admin4	adwqt33e	1
8	admin4	hdugeiud	1
9	customer3	jsdusaa4	0
10	user4	haihuf4	0

employee_ID	ename	DOB	salary	address	username	phone_number	position
1	pravallika	1990-05-15 00:00:00	65000	Potheri	_pravallika	3244	Manager
2	Reethika	1992-08-22 00:00:00	25000	Avadi	_Reethika	1234	Cashier
3	Avi	1996-11-10 00:00:00	30000	Tambaram	_Avi	4667	Cashier
4	Aarsh	1987-07-18 00:00:00	50000	Egmore	_Aarsh	3245	Supervisor
5	Abhi	1995-02-28 00:00:00	40000	Potheri	_Abhi	6768	Stock Clerk
6	Keerthi	1999-01-01 00:00:00	48000	Guduvancheri	_Keerthi	5452	Finance Coordinator
7	chandana	1990-01-10 00:00:00	50000	Potheri	_chandu	1634	Stock manager
8	tulasi	1989-06-28 00:00:00	30000	Puducherry	_tulasi	6895	Inventory clerk
9	veer	1995-10-15 00:00:00	25000	Banglore	_veer	9756	sales worker
10	Deeraj	1993-03-23 00:00:00	20000	Delhi	_deeraj	1734	janitorial staff

	eid	ename	gender	salary	department
▶	1	durga	female	20000	finance
	2	rekha	female	25000	finance
	3	moni	female	22000	finance
	4	reethu	female	40000	finance
	5	gauri	female	28000	finance
	6	raha	female	25000	finance
	7	chand	female	32000	finance
	8	sravani	female	35000	finance
	9	durga	female	10000	NULL
	10	John Doe	M	50000	finance

	eid	ename	gender	salary	department
▶	1	rahul	male	20000	finance
	2	goan	male	30000	finance
	3	sravan	male	28000	finance
	4	tony	male	40000	finance
	5	abhishek	male	30000	finance
	6	sanju	male	45000	finance
	7	revanth	male	48000	finance
	8	kasi	male	50000	finance

4.1.2 – In-Built Functions

	customerid	customername	gender	address
▶	105	chandu	female	avadi
	108	charvik	male	TNagar
*	NULL	NULL	NULL	NULL

	Totalcustomers
▶	10

	customerid	customername	gender	address
▶	105	chandu	female	avadi
	106	reena	female	potheri
	107	arush	male	tambaram
*	NULL	NULL	NULL	NULL

4.1.3 – Nested Queries

customerid	customername	gender	address
110	jayapal	male	Potheri
109	varsha	female	Puducherry
108	charvik	male	TNagar
107	arush	male	tambaram
106	reena	female	potheri

customerid	customername	gender	address
101	alex	male	Potheri
102	navya	female	Potheri
106	reena	female	potheri
110	jayapal	male	Potheri

billing_date	payment_method	total_amount	billing_ID
2024-03-15 00:00:00	Debit Card	125	6
2024-03-24 00:00:00	UPI	250	7
2024-04-02 00:00:00	UPI	450	8
2024-04-10 00:00:00	Credit Card	700	9
2024-04-15 00:00:00	Cash	1000	10

4.1.4 - Set Operations & Views

eid	ename	gender	salary	department
4	reethu	female	40000	finance
7	chand	female	32000	finance
8	sravani	female	35000	finance
10	John Doe	M	50000	finance
2	goan	male	30000	finance
4	tony	male	40000	finance
5	abhishek	male	30000	finance
6	sanju	male	45000	finance
7	revanth	male	48000	finance
8	kasi	male	50000	finance

eid	ename	gender	salary	department
2	rekha	female	25000	finance
4	reethu	female	40000	finance
6	raha	female	25000	finance
1	rahul	male	20000	finance
7	revanth	male	48000	finance

Views:

	eid	ename	gender	salary
▶	5	abhishek	male	30000
	2	goan	male	30000
	8	kasi	male	50000
	1	rahul	male	20000
	7	revanth	male	48000
	6	sanju	male	45000
	3	sravan	male	28000
	4	tony	male	40000

	count
▶	8

4.1.5 – PL/SQL Procedures & Functions

	total_records
▶	2

	TotalAmount
▶	3152.00

	sys.CountGender('Male')
▶	5

	sys.CountGender('Female')
▶	5

	sys.PrintAdmins()
▶	admin1,admin2,admin3,admin4,admin4

4.1.6 Cursors & Exception Handling

	v_result
▶	pravallika 65000, Reethika 25000, Avi 30000, A...

	Error: An error occurred during insertion
▶	Error: An error occurred during insertion

	Warning: A warning occurred while validating login
▶	Warning: A warning occurred while validating login

CHAPTER V
APPLICATIONS

The Grocery Management System (GMS) has various applications across different segments of the grocery retail industry:

1. **Independent Grocery Stores:** Small-scale grocery stores can utilize GMS to streamline their operations, including inventory management, sales tracking, and customer management. It helps them optimize inventory levels, improve customer service, and enhance overall efficiency.
2. **Supermarkets and Hypermarkets:** Large-scale supermarkets and hypermarkets can benefit from GMS to manage extensive product inventories, handle high volumes of transactions, and implement loyalty programs. It enables them to offer personalized services, analyze sales trends, and make data-driven decisions to enhance profitability.
3. **Online Grocery Retailers:** With the rise of e-commerce in the grocery sector, GMS can be integrated into online platforms to manage digital storefronts, track orders, and handle deliveries. It provides features for managing customer accounts, processing online payments, and coordinating logistics for efficient order fulfillment.
4. **Grocery Chains:** Chains operating multiple grocery stores can deploy GMS across their network to standardize processes, centralize data management, and implement uniform pricing and promotional strategies. It facilitates seamless communication between stores, headquarters, and suppliers for better coordination and decision-making.
5. **Wholesale and Distribution:** GMS can also be adapted for wholesalers and distributors supplying products to grocery retailers. It helps them manage inventory across multiple warehouses, track shipments, and coordinate deliveries to ensure timely supply to retail outlets.
6. **Specialty and Niche Markets:** Specialty grocery stores catering to specific dietary preferences, ethnic cuisines, or gourmet products can leverage GMS to manage unique inventory requirements, offer personalized recommendations to customers, and differentiate themselves in the market

CHAPTER VI
CONCLUSION

While developing this project we have learned a lot about HTML/CSS/JS/PHP/MySQL and working with database management, we have also learned how to make the application user-friendly (easy to use and handle) by hiding the complicated parts of it from the users.

During the development process, we studied carefully and understood the criteria for making software more demanding, we also realized the importance of maintaining a minimal margin for errors. I inferred various modules that should be used in the Grocery Management System which makes the flow much simpler. The importance of a database management system is that it's needed for developing efficient web applications, MySQL is the simplest and easiest to learn and implement. MySQL Workbench is an application that helps in developing MySQL codes and supports every functionality of SQL like views, PL/SQL procedures, functions, cursors, set operations, exception handling, etc.

BIBLIOGRAPHY

It has been a matter of immense pleasure, honor, and challenge to have this opportunity to take up this project and complete it successfully.

We have obtained information from various resources to design and implement our project.

We have acquired most of the knowledge from the Internet.

The following are some of the resources:

- https://ieeexplore.ieee.org/abstract/document/9028031/
- https://www.sciencedirect.com/science/article/pii/S1567422318300085
- https://www.academia.edu/download/49199190/SMART_GROCERY_MANAGEMENT_SYSTEM_USING_INTERNET_OF.pdf
- https://www.researchgate.net/profile/Sachin-Bhosale-4/publication/352477571_ONLINE_GROCERY_MANAGEMENT_SYSTEM/links/60caf0ca299bf1cd71d54e46/ONLINE-GROCERY-MANAGEMENT-SYSTEM.pdf
- https://www.tdcommons.org/dpubs_series/75/
- https://jespublication.com/uploads/2023-V14I9045.pdf
- https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/view/12156
- https://www.academia.edu/download/49433343/Grocery_Supply-Chain_Management_MyGROCER20161007-6480-6gv0xb.pdf
- https://www.emerald.com/insight/content/doi/10.1108/13673270810875903/full/html