➢ Objective of the Project

The main objective of the **Inventory Management System** project is to design and implement a structured database solution using **SQL** to manage and monitor product stock, supplier information, and purchase records efficiently.

Specifically, the project aims to:

1. **Create a normalized relational database** that stores information about products, suppliers, inventory levels, and purchase transactions.
2. **Demonstrate practical usage of SQL commands** such as:
    - **DDL (Data Definition Language)** to define and modify the structure of database objects.
    - **DML (Data Manipulation Language)** to manage and update data.
    - **DCL (Data Control Language)** to handle data access and security.

3. **Utilize aggregate functions** like SUM() and AVG() to generate business insights from transactional data.
4. **Implement JOIN operations** to retrieve meaningful and connected data across multiple tables.
5. **Ensure data consistency and integrity** through proper use of keys and constraints.
6. **Provide a scalable and real-time solution** that can help businesses keep track of their stock, purchases, and suppliers systematically.

❖ This project is intended as a foundation for understanding how database systems support real-world business operations by storing, retrieving, and analyzing data in an efficient and secure manner.

- Code

```
-- BHARAT INVENTORY MANAGEMENT SYSTEM
-- DDL (Data Definition Language)

-- 1. Create Tables

CREATE TABLE product (
    product_id INT PRIMARY KEY,
    product_name VARCHAR(100),
    category VARCHAR(50),
    price DECIMAL(10,2)
);

CREATE TABLE supplier (
    supplier_id INT PRIMARY KEY,
    supplier_name VARCHAR(100),
    contact_number VARCHAR(15),
    city VARCHAR(50)
);

CREATE TABLE inventory (
```

```
    product_id INT,
    quantity_available INT,
    FOREIGN KEY (product_id) REFERENCES
product(product_id)
);

CREATE TABLE purchase (
    purchase_id INT PRIMARY KEY,
    product_id INT,
    supplier_id INT,
    purchase_date DATE,
    quantity INT,
    FOREIGN KEY (product_id) REFERENCES
product(product_id),
    FOREIGN KEY (supplier_id) REFERENCES
supplier(supplier_id)
);

-- 2. Alter Tables

ALTER TABLE product ADD gst_percent DECIMAL(5,2);
ALTER TABLE supplier ADD email VARCHAR(100);
```

-- DML (Data Manipulation Language)

-- 1. Insert Data

```
INSERT INTO product VALUES (101, 'Dal', 'Grocery', 90.00, 5.00);
INSERT INTO product VALUES (102, 'Notebook', 'Stationery', 25.00, 12.00);
INSERT INTO product VALUES (103, 'Pencil', 'Stationery', 5.00, 18.00);

INSERT INTO supplier VALUES (201, 'Ramesh Traders', '9876543210', 'Mumbai', 'ramesh@traders.in');
INSERT INTO supplier VALUES (202, 'Suresh Wholesalers', '9123456789', 'Delhi', 'suresh@wholesale.in');

INSERT INTO inventory VALUES (101, 100);
INSERT INTO inventory VALUES (102, 200);
INSERT INTO inventory VALUES (103, 300);
```

```
INSERT INTO purchase VALUES (301, 101, 201, '2024-04-01', 50);

INSERT INTO purchase VALUES (302, 102, 202, '2024-04-05', 100);

INSERT INTO purchase VALUES (303, 103, 201, '2024-04-10', 150);

-- 2. Update Data

UPDATE product SET price = 95.00 WHERE product_id = 101;

UPDATE inventory SET quantity_available = 180 WHERE product_id = 103;

-- DCL (Data Control Language)

-- 1. Grant Permissions

GRANT SELECT, INSERT ON product TO 'inventory_user';

GRANT UPDATE ON inventory TO 'inventory_user';

-- 2. Revoke Permissions
```

```
REVOKE INSERT ON product FROM 'inventory_user';
REVOKE UPDATE ON inventory FROM 'inventory_user';

--  AGGREGATE FUNCTIONS

-- 1. Total quantity purchased per product
SELECT product_id, SUM(quantity) AS total_purchased
FROM purchase
GROUP BY product_id;

-- 2. Average price of products in each category
SELECT category, AVG(price) AS avg_price
FROM product
GROUP BY category;

-- JOIN QUERIES

-- 1. Show all products with current inventory status
SELECT p.product_name, p.category, i.quantity_available
FROM product p
JOIN inventory i ON p.product_id = i.product_id;
```

```
-- 2. Show purchase records with product and supplier details
SELECT
    pr.purchase_id,
    p.product_name,
    s.supplier_name,
    s.city,
    pr.purchase_date,
    pr.quantity
FROM purchase pr
JOIN product p ON pr.product_id = p.product_id
JOIN supplier s ON pr.supplier_id = s.supplier_id;

-- 3. List all suppliers and the products they supplied
SELECT DISTINCT s.supplier_name, p.product_name
FROM supplier s
JOIN purchase pr ON s.supplier_id = pr.supplier_id
JOIN product p ON pr.product_id = p.product_id;

-- 4. Join with aggregate: Total quantity supplied by each
supplier
```

```
SELECT
    s.supplier_name,
    SUM(pr.quantity) AS total_supplied
FROM supplier s
JOIN purchase pr ON s.supplier_id = pr.supplier_id
GROUP BY s.supplier_name;
```

- Output

1. Total quantity purchased per product

```
SELECT product_id, SUM(quantity) AS
total_purchased
FROM purchase
GROUP BY product_id;
```

product_id	total_purchased
101	50
102	100
103	150

2. Average price per category

sql

```
SELECT category, AVG(price) AS avg_price
FROM product
GROUP BY category;
```

category	avg_price
Grocery	95.00
Stationery	15.00

3. Products with Inventory Status

sql

- SELECT p.product_name, p.category, i.quantity_available
- FROM product p
- JOIN inventory i ON p.product_id = i.product_id;

product_name	category	quantity_available
Dal	Grocery	100

product_name	category	quantity_available
Notebook	Stationery	200
Pencil	Stationery	180

. Purchases with Product and Supplier Details

sql

```
SELECT
pr.purchase_id,
p.product_name,
s.supplier_name,
s.city,
pr.purchase_date,
```

purchase_id	product_name	supplier_name	city	purchase_date	quantity
301	Dal	Ramesh Traders	Mumbai	2024-04-01	50
302	Notebook	Suresh Wholesalers	Delhi	2024-04-05	100
303	Pencil	Ramesh Traders	Mumbai	2024-04-10	150

- pr.quantity
- FROM purchase pr
- JOIN product p ON pr.product_id = p.product_id
- JOIN supplier s ON pr.supplier_id = s.supplier_id;

5. List all suppliers and the products they supplied

- SELECT DISTINCT s.supplier_name, p.product_name

- FROM supplier s
- JOIN purchase pr ON s.supplier_id = pr.supplier_id
- JOIN product p ON pr.product_id = p.product_id;

supplier_name	product_name
Ramesh Traders	Dal
Suresh Wholesalers	Notebook
Ramesh Traders	Pencil

6. Total quantity supplied by each supplier

SELECT

- s.supplier_name,
- SUM(pr.quantity) AS total_supplied
- FROM supplier s

- JOIN purchase pr ON s.supplier_id = pr.supplier_id
- GROUP BY s.supplier_name;

• **supplier_name**	• **total_supplied**
• Ramesh Traders	• 200
• Suresh Wholesalers	• 100

❖ Conclusion

❖ The **Inventory Management System** project successfully demonstrates the implementation of core **Database Management System (DBMS)** concepts using SQL. The system consists of four interrelated tables: product, supplier, inventory, and purchase, representing a realistic model of a small-to-medium scale business inventory.

Throughout the project:

- **Data Definition Language (DDL)** was used to **create and alter** the database schema.

- **Data Manipulation Language (DML)** handled **inserting and updating** product, supplier, and purchase information.
- **Data Control Language (DCL)** was applied to **grant and revoke access**, showcasing basic security control in SQL.
- **Aggregate functions** like SUM() and AVG() provided meaningful business insights such as total purchases and average prices.
- **JOIN operations** were used to fetch and combine related data across multiple tables, demonstrating how normalized relational databases work in practice.
- The project highlights the importance of database normalization, efficient querying, and proper access control in real-world applications. It provides a scalable foundation for adding further functionality such as sales tracking, low stock alerts, or supplier ratings.
- Overall, this DBMS project serves as a comprehensive, real-world simulation of managing

inventory in an organized, secure, and insightful manner using SQL.