**Shopping Mall Management System**

**Cover Page**

* Project Title: Mall Management System
* Student Name: P.Kanagalakshmi
* Course/Batch: DataAnalysis
* Date:

**Table of Contents**

1. Introduction
2. Objectives
3. System Design

* 3.1 SQL Database Design
* 3.2 Python Integration

1. Implementation

* 4.1 SQL Implementation
* 4.2 Python Implementation

1. Sample Queries and Analytics

* 5.1 SQL Queries
* 5.2 Python Analytics

1. Results
2. Conclusion
3. Future Enhancements

**1. Introduction**

The Mall Management System automates the management of malls, stores, customers, and purchases. The system uses a **MySQL database** to store structured data and **Python analytics** to generate insights on sales, revenue, and customer behavior.

**2. Objectives**

* Manage mall, store, and customer information efficiently.
* Track customer purchases and store revenue.
* Provide detailed analytics for sales and revenue.
* Generate visual reports using Python for better decision-making.
* Maintain audit logs for critical data operations.

**3. System Design**

**3.1 SQL Database Design**

**Tables:**

* **mall** – Stores mall details (name, code, city).
* **store** – Stores store details linked to malls.
* **customer** – Stores customer information and associated store.
* **purchase** – Tracks customer purchases in stores.
* **mall\_audit** – Maintains audit logs for insert/update/delete actions on malls.

**Triggers & Procedures:**

* **mall\_after\_insert** – Logs insert actions in mall\_audit.
* **mall\_after\_update** – Logs update actions in mall\_audit.
* **mall\_after\_delete** – Logs delete actions in mall\_audit.

**Views:**

**mall\_store\_customer\_details** – Combines mall, store, and customer information for analytics.

**3.2 Python Integration**

* Python connects to MySQL using PyMySQL.
* Pandas & NumPy are used for data manipulation and aggregation.
* Matplotlib creates visualizations of revenue, top stores, and customer spending trends.

**4. Implementation**

**4.1 SQL Implementation**

* Created the database mall\_project with tables, triggers, and sample data.
* Triggers maintain audit logs for mall insert/update/delete actions.
* Views simplify combined analysis of malls, stores, and customers.
* Sample aggregate queries calculate sales per store, revenue per mall, and top customers.

**4.2 Python Implementation**

import pymysql

import pandas as pd

import matplotlib.pyplot as plt

connection = pymysql.connect(

host="localhost",

user="root",

password="susi123",

database="mall\_project"

)

**# Load purchase data**

purchase\_df = pd.read\_sql("SELECT \* FROM purchase", connection)

store\_df = pd.read\_sql("SELECT \* FROM store", connection)

**# Merge data for analytics**

df = pd.merge(purchase\_df, store\_df, left\_on='s\_id', right\_on='id')

**# Total sales per store**

sales\_per\_store = df.groupby('store\_name')['amount'].sum().sort\_values(ascending=False)

sales\_per\_store.plot(kind='bar', figsize=(10,5))

plt.ylabel("Total Sales")

plt.title("Total Sales per Store")

plt.show()

**5. Sample Queries and Analytics**

**5.1 SQL Queries**

* **Total revenue per mall:**

SELECT m.mall\_name, SUM(p.amount) AS total\_revenue

FROM purchase p

JOIN store s ON p.s\_id = s.id

JOIN mall m ON s.m\_id = m.id

GROUP BY m.mall\_name;

* **Top 5 customers by spending:**

SELECT c.cust\_name, SUM(p.amount) AS total\_spent

FROM purchase p

JOIN customer c ON p.cust\_id = c.id

GROUP BY c.cust\_name

ORDER BY total\_spent DESC

LIMIT 5;

* **Customers per store category:**

SELECT s.category, COUNT(c.id) AS total\_customers

FROM customer c

JOIN store s ON c.s\_id = s.id

GROUP BY s.category;

**5.2 Python Analytics**

* Total sales per store – bar chart visualization.
* Revenue distribution by category – pie chart.
* Top-spending customers – horizontal bar chart.
* Trend analysis of purchases over time – line chart.

**6. Results**

* Complete mall, store, and customer data management.
* Automated revenue and sales reporting.
* Visual insights into top-performing stores and malls.
* Audit trails for mall modifications.

**7. Future Enhancements**

* Mobile application for customer engagement.
* Online payment and loyalty points tracking.
* Predictive analytics for sales and store performance.
* Real-time dashboards for mall management.

**8. Conclusion**

The Mall Management System efficiently integrates **SQL database operations** and **Python analytics**. SQL manages structured data, constraints, and triggers, while Python provides analytical insights and visualizations. The system improves decision-making and sales tracking for mall management.