A

Project Report

on

“LifeStyle Store Management System”

submitted to

SavitriBai Phule Pune University

in Partial fulfillment of

TE Computer Engineering, SEM I

for the Academic Year

2022-23

Submitted by

**Sahil Apture(CO3002)**

**Basant Bhagat(CO3007)**

**Atharva Khurpe(CO3004)**

Under the guidance of

Prof. Ruchira Tare



Department of Computer Engineering

Trinity College of Engineering and Research

**INDEX**

1. Brief Overview of Project
2. Software and Hardware requirement
3. Theory
4. Advantages of Project
5. Limitations of Project
6. Steps for connecting Database
7. Source Code of Project
8. Output Screens
9. Future Enhancement of Project
10. Bibliography

**Database Management Systems Mini-project**

**Title:** Lifestyle Store Management System

**Group:** 07

**Submitted By:**

**Software Requirements:** MySQL, Python 3.9, MySQL Connector

**Hardware Requirements:** Any CPU with Pentium Processor and 1 ………………………………………...GB Hard Disk

**Database:** MySQL

**Brief Overview of Project:**

The main objective of the python project on fashion store management is to manage the details of sales, discounts, payments, products, and inventory digitally.

The project is totally built at the administrative end and only the administrator is guaranteed the access.

The purpose of the project is to build an application program to reduce the manual work for managing the sales, discounts, stock, and payments.

It tracks all the details about stocks, products, and inventory; it also prints various reports as per input given by the user.

***INPUT DATA AND VALIDATION OF PROJECT***

1. All the fields such as sales payments discounts are validated and do not take invalid values.
2. Each form of sales, discounts, stock cannot accept the blank values.
3. Avoiding errors in data.
4. Controlling amount of input.

**SOFTWARE AND HARDWARE REQUIREMENTS:**

Data file handling has been effectively used in the program. The database is a collection of interrelated data to serve multiple applications. That is database programs create files of information. So we see that files are worked with most, inside the program.

DBMS: The software required for the management of data is called DBMS. It has 3 models:

• Relation model

• Hierarchical model

• Network model

*RELATIONAL MODEL:* It’s based on the concept of relation. Relation is the table that consists of rows and columns. The rows of the table are called tuples and the columns of the table are called attributes. The Number of rows in the table is called cardinality. Number of columns in the table is called the degree.

*HIERARCHICAL MODEL:* In this type of model, we have multiple records for each record. A particular record has one parent record. No child record can exist without a parent record. In this, the records are organized in trees.

*NETWORK MODEL:* In this, the data is represented by collection of records and relationship is represented by (ink or association.

*CHARACTERISTICS OF DBMS:*

• It reduces the redundancy

• Reduction of data in inconsistency

• Data sharing

• Data standardization

DIFFERENT TYPES OF FILES: -BASED ON ACCESS:

• Sequential file

• Serial file

• Random (direct access) file BASED ON STORAGE:-

• Text file

• Binary File

**NEED OF COMPUTERISATION**

Over the decades computers and fashion have developed gradually, changed with time, taste and trend. But nobody knew that a time would come when both these fields would complement each other so well. Today fashion design has reached new heights by computer aided methods of design. As a result of which, the computer industry has got its new customer. Computer technology is making waves in the fashion design zone. From determining textile weaves to sizing designs; computers are a vital component of the fashion industry. Computer aided design (CAD) programs reduce the demand for manual sketches. New software programs continue to replace old manual skills. Going by the wayside are "old fashioned" flat pattern construction, pencil sketching and traditional math-based pattern sizing. Those who lag in math and falter at sketching can now breathe a little easier. Manually figuring size adjustments and cutting pattern pieces instils that knowledge. Software programs constantly evolve. A program used today may be obsolete within several years. Being trained on today's software does not guarantee it will be used when you are ready to go out into the field. Understanding calculations is timeless, as is computer competency. Software, however, shifts rapidly.

**Advantages of project**

1. It generates reports on sales, discounts and stocks.
2. Provides filter reports on payments, inventory and products.
3. We can easily export PDF on sales, products and stocks.
4. Applications can also provide excel export for sales and discounts.
5. It deals with monitoring the information and transaction of products.
6. It increases the efficiency of managing sales and discounts.
7. It has higher efficiency of editing, adding and updating of records.
8. Provides the searching facilities on various factors.

**Limitations**

1. Excel export has not been developed for stocks and products.
2. The transactions are executed in offline mode only.
3. Online transactions for sales, discounts, or other data modifications are not possible.
4. Offline reports of sales, products, discounts and stocks cannot be generated due to batch mode execution.

**SQL (Structured Query Language)**

It is used to perform operations on the records stored in the database, such as updating records, inserting records, deleting records, creating and modifying database tables, views, etc.

SQL is not a database system, but it is a query language.

Suppose you want to perform the queries of SQL language on the stored data in the database. You are required to install any database management system in your systems, for example, [Oracl](https://www.javatpoint.com/oracle-tutorial)e, [MySQL](https://www.javatpoint.com/mysql-tutorial), [MongoDB](https://www.javatpoint.com/mongodb-tutorial), [PostgreSQL](https://www.javatpoint.com/postgresql-tutorial), [SQL Server](https://www.javatpoint.com/sql-server-tutorial), [DB2](https://www.javatpoint.com/db2-tutorial), etc.

We have made use of MySQL for this project.

## What is SQL?

SQL is a short-form of the structured query language, and it is pronounced as S-Q-L or sometimes as See-Quell.

This database language is mainly designed for maintaining the data in relational database management systems. It is a special tool used by data professionals for handling structured data (data which is stored in the form of tables). It is also designed for stream processing in RDSMS.

You can easily create and manipulate the database, access and modify the table rows and columns, etc. This query language became the standard of ANSI in the year of 1986 and ISO in the year of 1987.

If you want to get a job in the field of data science, then it is the most important query language to learn. Big enterprises like Facebook, Instagram, and LinkedIn, use SQL for storing the data in the back-end.

## Why SQL?

Nowadays, SQL is widely used in data science and analytics. Following are the reasons which explain why it is widely used:

* The basic use of SQL for data professionals and SQL users is to insert, update, and delete the data from the relational database.
* SQL allows the data professionals and users to retrieve the data from the relational database management systems.
* It also helps them to describe the structured data.
* It allows SQL users to create, drop, and manipulate the database and its tables.
* It also helps in creating the view, stored procedure, and functions in the relational database.
* It allows you to define the data and modify that stored data in the relational database.
* It also allows SQL users to set the permissions or constraints on table columns, views, and stored procedures.

## History of SQL

"A Relational Model of Data for Large Shared Data Banks" was a paper which was published by the great computer scientist "E.F. Codd" in 1970.

The IBM researchers Raymond Boyce and Donald Chamberlin originally developed the SEQUEL (Structured English Query Language) after learning from the paper given by E.F. Codd. They both developed the SQL at the San Jose Research laboratory of IBM Corporation in 1970.

At the end of the 1970s, relational software Inc. developed their own first SQL using the concepts of E.F. Codd, Raymond Boyce, and Donald Chamberlin. This SQL was totally based on RDBMS. Relational Software Inc., which is now known as Oracle Corporation, introduced the Oracle V2 in June 1979, which is the first implementation of SQL language. This Oracle V2 version operates on VAX computers.

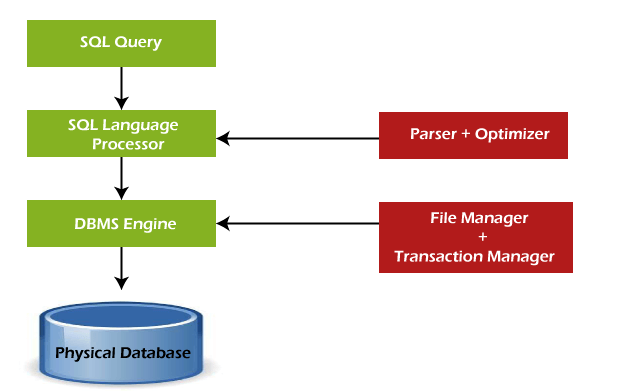
## Process of SQL

When we are executing the command of SQL on any Relational database management system, then the system automatically finds the best routine to carry out our request, and the SQL engine determines how to interpret that particular command.

Structured Query Language contains the following four components in its process:

* Query Dispatcher
* Optimization Engines
* Classic Query Engine
* SQL Query Engine, etc.

A classic query engine allows data professionals and users to maintain non-SQL queries. The architecture of SQL is shown in the following diagram:



## Some SQL Commands

The SQL commands help in creating and managing the database. The most common SQL commands which are highly used are mentioned below:

1. CREATE command
2. UPDATE command
3. DELETE command
4. SELECT command
5. DROP command
6. INSERT command

### CREATE Command

This command helps in creating the new database, new table, table view, and other objects of the database.

### UPDATE Command

This command helps in updating or changing the stored data in the database.

### DELETE Command

This command helps in removing or erasing the saved records from the database tables. It erases single or multiple tuples from the tables of the database.

### SELECT Command

This command helps in accessing the single or multiple rows from one or multiple tables of the database. We can also use this command with the WHERE clause.

### DROP Command

This command helps in deleting the entire table, table view, and other objects from the database.

### INSERT Command

This command helps in inserting the data or records into the database tables. We can easily insert the records in single as well as multiple rows of the table.

## Advantages of using SQL:

SQL provides various advantages which make it more popular in the field of data science. It is a perfect query language which allows data professionals and users to communicate with the database. Following are the best advantages or benefits of Structured Query Language:

**1. No programming needed**

SQL does not require a large number of coding lines for managing the database systems. We can easily access and maintain the database by using simple SQL syntactical rules. These simple rules make the SQL user-friendly.

**2. High-Speed Query Processing**

A large amount of data is accessed quickly and efficiently from the database by using SQL queries. Insertion, deletion, and updation operations on data are also performed in less time.

**3. Standardized Language**

SQL follows the long-established standards of ISO and ANSI, which offer a uniform platform across the globe to all its users.

**4. Portability**

The structured query language can be easily used in desktop computers, laptops, tablets, and even smartphones. It can also be used with other applications according to the user's requirements.

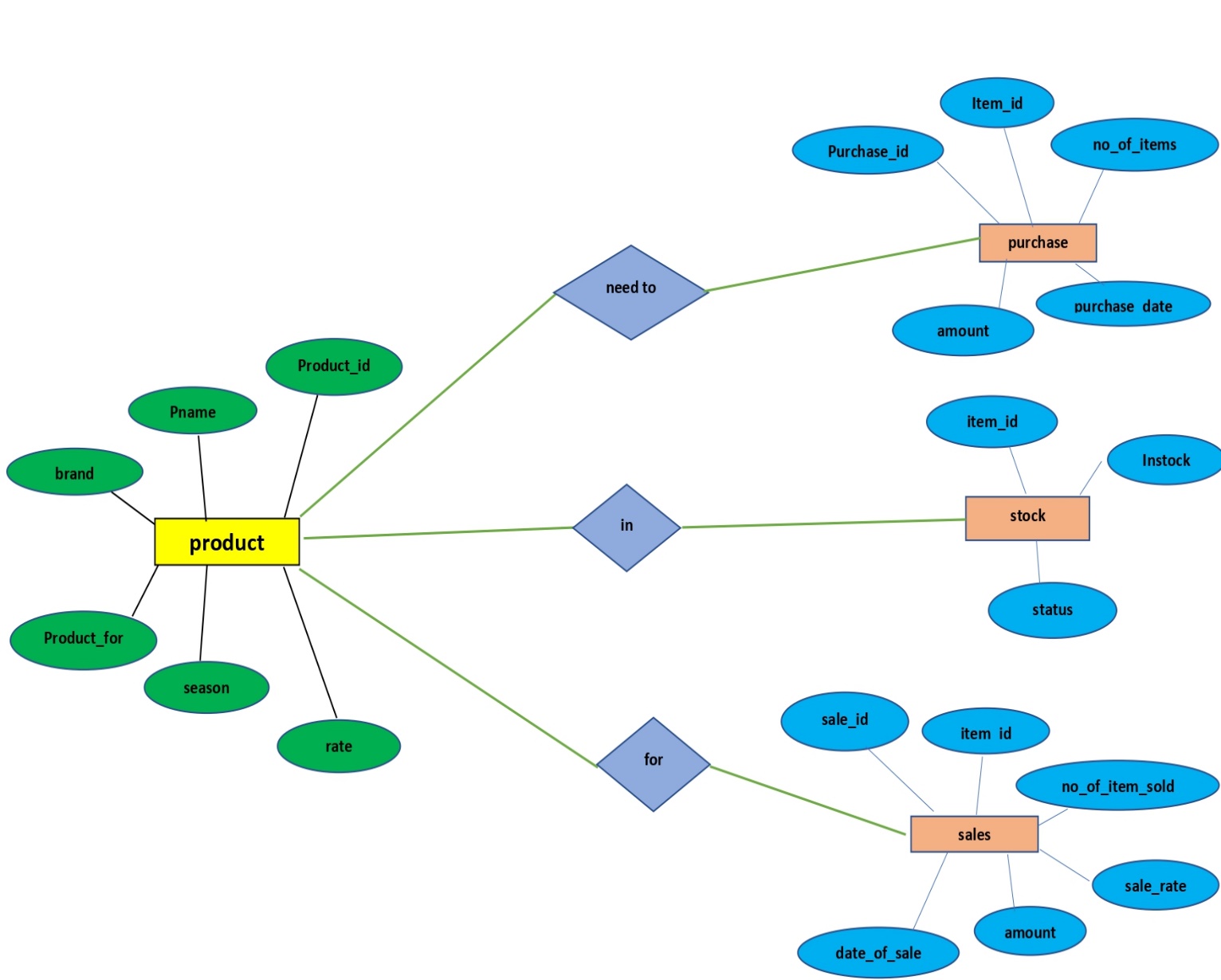
**5. Interactive language**

We can easily learn and understand the SQL language. We can also use this language for communicating with the database because it is a simple query language. This language is also used for receiving the answers to complex queries in a few seconds.

**6. More than one Data View**

The SQL language also helps in making the multiple views of the database structure for the different database users.

***ER DIAGRAM:***



**Source Code screening**

DBMS: MySQL

Host: local host

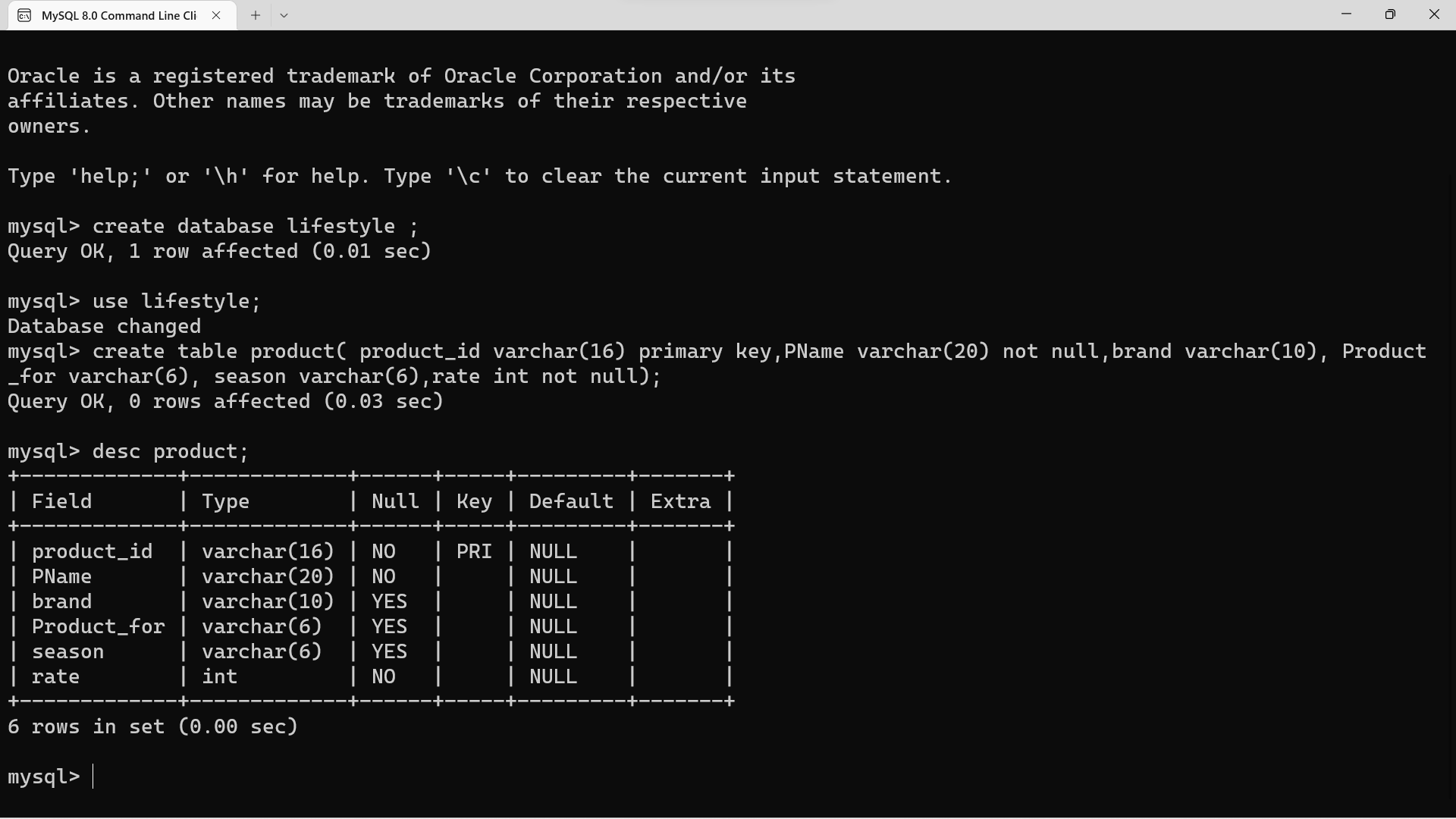
User: root

Pass: root

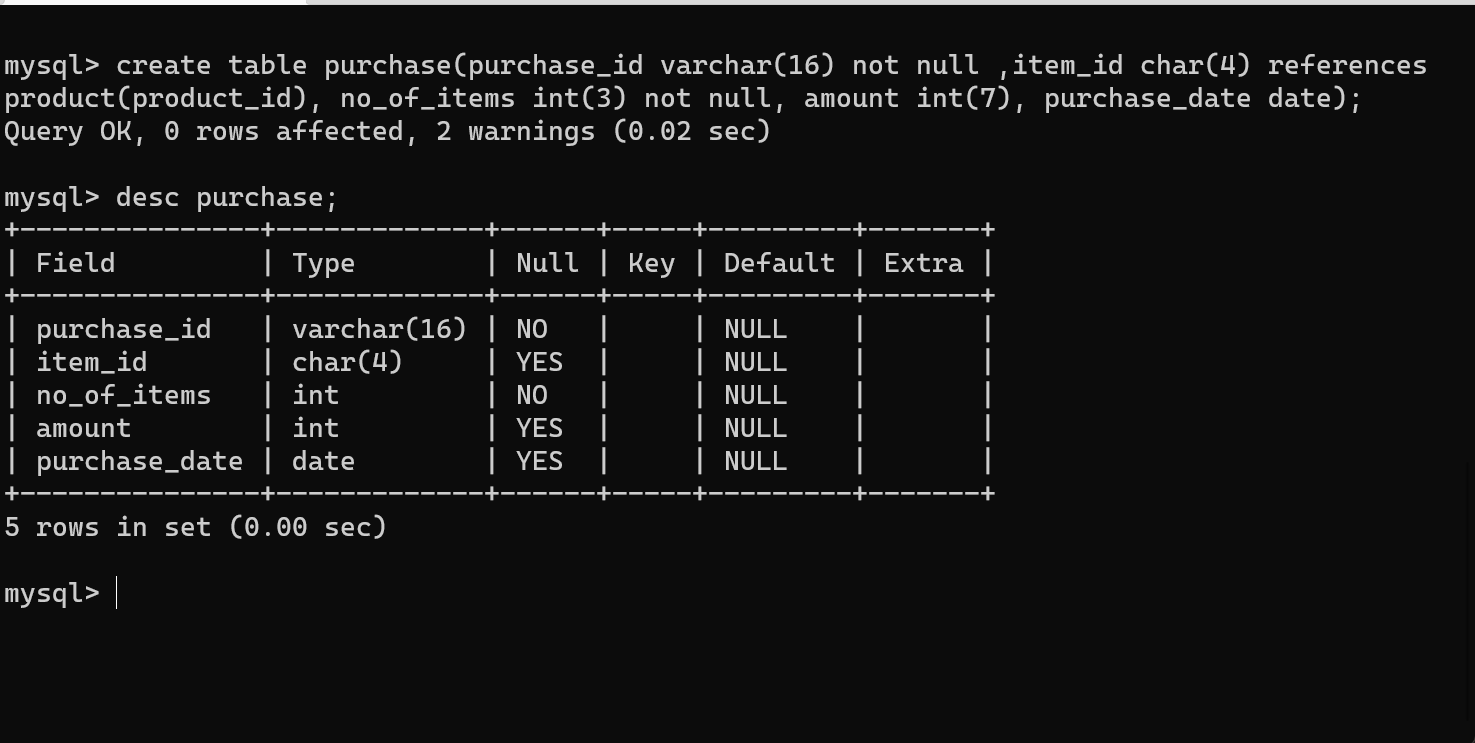
Database: lifestyle

Table Structure: (Images Bellow)

1. *Product table*



1. *Purchase table*

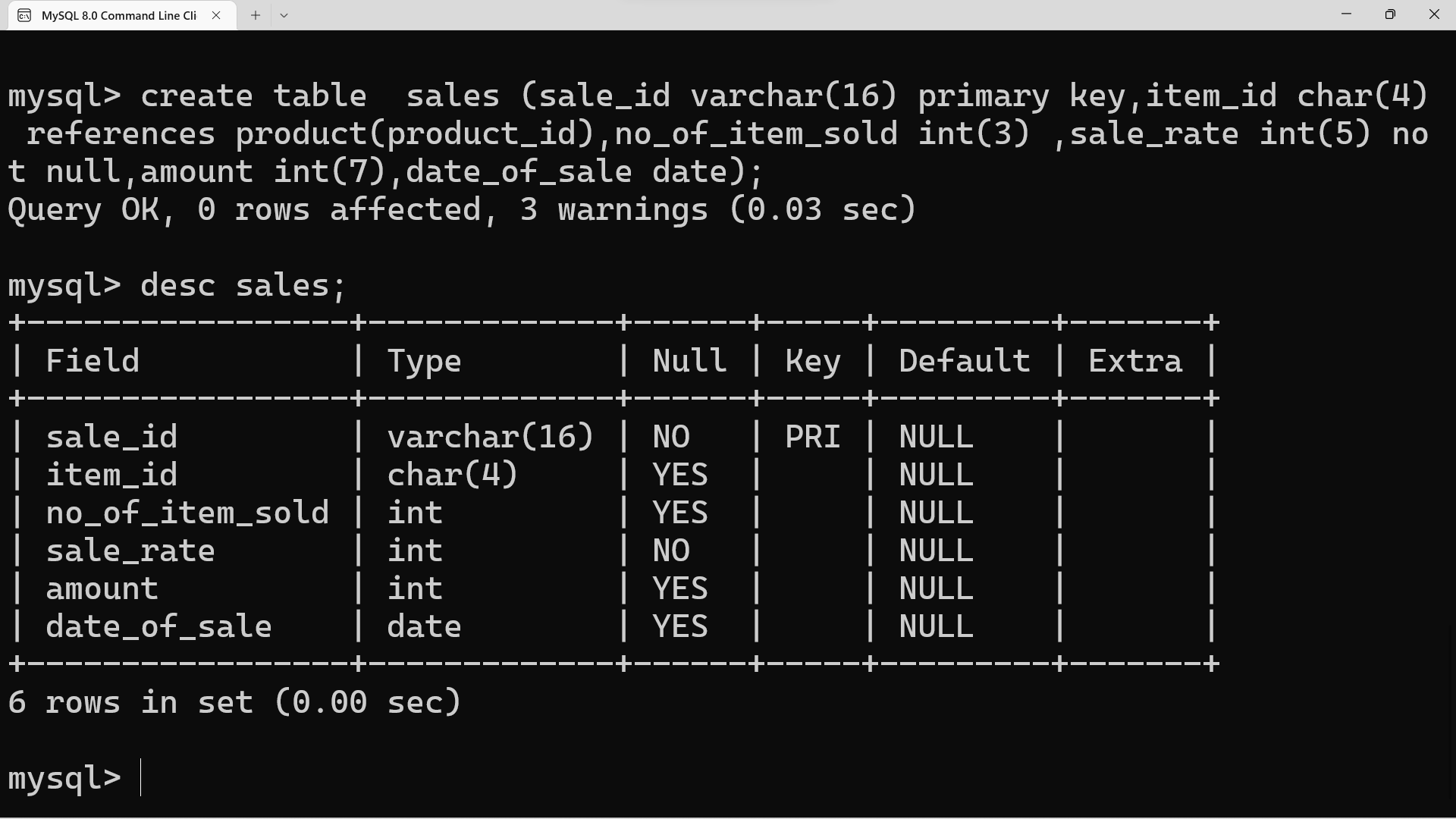
Note: In Purchase table take the purchase ID as varchar (16)

1. *Stock table*

Text

Description automatically generated

1. *Sales table*



Note: In Purchase table you can take the sale\_id as varchar (16)

**Steps for connection of Database:**

To be able to implement the code , you should have MySQL installed on your computer.

You can download a free MySQL database at <https://www.mysql.com/downloads/>.

## Install MySQL Driver

Python needs a MySQL driver to access the MySQL database.

In this tutorial we will use the driver "MySQL Connector".

We recommend that you use PIP to install "MySQL Connector".

PIP is most likely already installed in your Python environment.

Navigate your command line to the location of PIP, and type the following:

**Download and install “MySQL Connector”:**

C:\Users\Your Name \AppData\Local\Programs\Python\Python36-32\

Scripts>python -m pip install mysql-connector-

*You have downloaded and installed a MySQL driver.*

## Test MySQL Connector

To test if the installation was successful, or if you already have "MySQL Connector" installed, create a Python page with the following content:

demo\_mysql\_test.py:

import mysql.connector

If the above code was executed with no errors, "MySQL Connector" is installed and ready to be used.

## Create Connection

Start by creating a connection to the database.

Use the username and password from your MySQL database:

*demo\_mysql\_connection.py:*

import mysql.connector

mydb = mysql.connector.connect(

host="localhost",

user="*yourusername*",

password="*yourpassword*")

print(mydb)

Now you can start querying the database using SQL statements.

**SQL commands to create both the tables:**

create database lifestyle ;

use lifestyle;

create table product( product\_id varchar(16) primary key,PName varchar(20) not null,brand varchar(10), Product\_for varchar(6), season varchar(6),rate int not null);

create table purchase(purchase\_id varchar(16) not null ,item\_id char(4) references product(product\_id), no\_of\_items int(3) not null, amount int(7), purchase\_date date);

create table stock (item\_id char(4) references product(product\_id),Instock int(3) not null,status varchar(10) not null);

create table sales (sale\_id varchar(16) primary key,item\_id char(4) references product(product\_id),no\_of\_item\_sold int(3) ,sale\_rate int(5) not null,amount int(7),date\_of\_sale date);

**Python Source code:**

import os

import platform

import mysql.connector

import pandas as pd

import datetime

mydb=mysql.connector.connect(host="localhost",\

user="root",\

passwd="root",\

database="lifestyle")

mycursor=mydb.cursor()

def AddProduct():

L=[]

stk=[]

pid=input("Enter the Product ID : ")

L.append(pid)

IName=input("Enter the Product Name : ")

L.append(IName)

brnd=input("Enter the Product Brand Name : ")

L.append(brnd)

fr=input("Enter Male/Female/Kids : ")

L.append(fr)

sn=input("Enter Winter/Summer : ")

L.append(sn)

rate=int(input("Enter the Rates for Product :"))

L.append(rate)

product=(L)

sql="Insert into product (product\_id,PName,brand,Product\_for,Season,rate)values(%s,%s,%s,%s,%s,%s)"

mycursor.execute(sql,product)

mydb.commit()

stk.append(pid)

stk.append(0)

stk.append("No")

st=(stk)

sql="insert into stock(item\_id, Instock, status) values(%s,%s,%s)"

mycursor.execute(sql,st)

mydb.commit()

print("One Product inserted ")

def EditProduct():

pid=input("Enter product ID to be edited : ")

sql="select \* from product where product\_id=%s"

ed=(pid,)

mycursor.execute(sql,ed)

res=mycursor.fetchall()

for x in res:

print(x)

print("")

fld=input("Enter the field which you want to edit : ")

val=input("Enter the value you want to set : ")

sql="Update product set " + fld +"='" + val + "' where product\_id='" + pid + "'"

sq=sql

mycursor.execute(sql)

print("Editing Done : ")

print("After correction the record is : ")

sql="select \* from product where product\_id=%s"

ed=(pid,)

mycursor.execute(sql,ed)

res=mycursor.fetchall()

for x in res:

print(x)

mydb.commit()

def DelProduct():

pid=input("Enter the Product)id to be deleted : ")

sql="delete from sales where item\_id=%s"

id=(pid,)

mycursor.execute(sql,id)

mydb.commit()

sql="delete from purchase where item\_id=%s"

mycursor.execute(sql,id)

mydb.commit()

sql="delete from stock where item\_id=%s"

mycursor.execute(sql,id)

mydb.commit()

sql="delete from product where product\_id=%s"

mycursor.execute(sql,id)

mydb.commit()

print("One Item Deleted")

def ViewProduct():

print("Display Menu: Select the category to display the data")

print("1. All Details")

print("2. Product Name:")

print("3. Product Brand:")

print("4. Product For:")

print("5. Product Season:")

print("6. Product ID:")

x=0

ch=int(input("Enter your choice to display : "))

if ch==1:

sql="select \* from product"

mycursor.execute(sql)

res=mycursor.fetchall()

for x in res:

print(x)

x=1

elif ch==2:

var='PName'

val=input("Enter the name of Product : ")

elif ch==3:

var='brand'

val=input("Enter the name of Brand : ")

elif ch==4:

var='Product\_for'

val=input("Enter Male/Femal/Kids : ")

elif ch==5:

var='season'

val=input("Enter the Season : ")

elif ch==6:

var='product\_id'

val=input("Enter the Product\_id : ")

if x==0:

sql="select \* from product where " + var + " = %s"

sq=sql

tp=(val,)

mycursor.execute(sq,tp)

res=mycursor.fetchall()

for x in res:

print(x)

def PurchaseProduct():

mn=""

dy=""

now=datetime.datetime.now()

purchaseID="P"+str(now.year)+str(now.month)+str(now.day)+str(now.hour)+str(now.minute)+str(now.second)

L=[]

Lst=[]

L.append(purchaseID)

itemId=input("Enter Product ID : ")

L.append(itemId)

itemNo=int(input("Enter the number of Items : "))

L.append(itemNo)

sql="select rate from product where product\_id=%s"

pid=(itemId,)

mycursor.execute(sql,pid)

res=mycursor.fetchone()

for x in res:

print("rate is : ", x)

amount=x\*itemNo

print("Amount is :", amount)

L.append(amount)

mnth=now.month

if mnth<=9:

mn="0"+str(mnth)

else:

mn=str(mnth)

day=now.day

if day<=9:

dy="0"+str(day)

else:

dy=str(day)

dt=str(now.year)+"-"+mn+"-"+dy

L.append(dt)

tp=(L)

sql="insert into purchase(purchase\_id,item\_id,no\_of\_items,amount,Purchase\_date)values(%s,%s,%s,%s,%s)"

mycursor.execute(sql,tp)

mydb.commit()

sql="Select Instock from stock where item\_id=%s"

mycursor.execute(sql,pid)

res=mycursor.fetchall()

status="No"

for x in res:

print(x)

instock=x[0]+itemNo

if instock>0:

status="Yes"

Lst.append(instock)

Lst.append(status)

Lst.append(itemId)

tp=(Lst)

sql="update stock set instock=%s,status=%s where item\_id=%s"

mycursor.execute(sql,tp)

mydb.commit()

print("1 Item purchased and saved in Database")

def ViewPurchase():

item=input("Enter Product Name : ")

sql="select product.product\_id, product.PName,product.brand,purchase.no\_of\_items,purchase.purchase\_date, purchase.amount from product INNER JOIN purchase ON product.product\_id=purchase.item\_id and product.PName=%s"

itm=(item,)

mycursor.execute(sql,itm)

res=mycursor.fetchall()

for x in res:

print(x)

def ViewStock():

item=input("Enter Product Name : ")

sql="select product.product\_id,product.PName,stock.Instock,\

stock.status from stock, product where \

product.product\_id=stock.item\_id and product.PName=%s"

itm=(item,)

mycursor.execute(sql,itm)

res=mycursor.fetchall()

for x in res:

print(x)

def SaleProduct():

now=datetime.datetime.now()

saleID="S"+str(now.year)+str(now.month)+str(now.day)+str(now.hour)+str(now.minute)+str(now.second)

L=[]

L.append(saleID)

itemId=input("Enter Product ID : ")

L.append(itemId)

itemNo=int(input("Enter the number of Items : "))

L.append(itemNo)

sql="select rate from product where product\_id=%s"

pid=(itemId,)

mycursor.execute(sql,pid)

res=mycursor.fetchall()

for x in res:

print("The rate of item is :",x)

dis=int(input("Enter the discount : "))

saleRate=x[0]-(x[0]\*dis/100)

L.append(saleRate)

amount=itemNo\*saleRate

L.append(amount)

mnth=now.month

if mnth<=9:

mn="0"+str(mnth)

else:

mn=str(mnth)

day=now.day

if day<=9:

dy="0"+str(day)

else:

dy=str(day)

dt=str(now.year)+"-"+mn+"-"+dy

L.append(dt)

tp=(L)

sql="insert into sales (sale\_id, item\_id,no\_of\_item\_sold,\

sale\_rate,amount,date\_of\_sale) values(%s,%s,%s,%s,%s,%s)"

mycursor.execute(sql,tp)

mydb.commit()

sql="Select Instock from stock where item\_id=%s"

mycursor.execute(sql,pid)

res=mycursor.fetchall()

for x in res:

print("Total Items in Stock are : ",x)

instock=x[0]-itemNo

if instock>0:

status="Yes"

else:

status="No"

tp=(instock,status,itemId)

sql="update stock set instock=%s,status=%s where item\_id=%s"

print("Remaining Items in Stock are : ",instock)

mycursor.execute(sql,tp)

mydb.commit()

def ViewSales():

item=input("Enter Product Name : ")

sql="select product.product\_id, product.PName,product.brand,\

sales.no\_of\_item\_sold,sales.date\_of\_sale,sales.amount \

from sales, product where product.product\_id=sales.item\_id \

and product.PName=%s"

itm=(item,)

mycursor.execute(sql,itm)

res=mycursor.fetchall()

for x in res:

print(x)

def MenuSet(): #Function For The SFashion Store System

print("\*"\*80)

print("\* \* \* \* \* \* \* Welcome to the Lifestyle Store \* \* \* \* \* \* \* ")

print("\* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \* \*")

print("\*"\*80)

print("")

print("Enter 1 : To Add Product ")

print("Enter 2 : To Edit Product ")

print("Enter 3 : To Delete Product ")

print("Enter 4 : To View Product ")

print("Enter 5 : To Purchase Product")

print("Enter 6 : To View Purchases")

print("Enter 7 : To View Stock Detials")

print("Enter 8 : To put an item on Sale")

print("Enter 9 : To View Sales Details")

try: #Using Exceptions For Validation

userInput = int(input("Please Select An Above Option: ")) #Will Take Input From User

except ValueError:

exit("\nHy! That's Not A Number") #Error Message

else:

print("\n") #Print New Line

if(userInput == 1):

AddProduct()

elif(userInput == 2):

EditProduct()

elif (userInput==3):

DelProduct()

elif (userInput==4):

ViewProduct()

elif (userInput==5):

PurchaseProduct()

elif (userInput==6):

ViewPurchase()

elif (userInput==7):

ViewStock()

elif (userInput==8):

SaleProduct()

elif (userInput==9):

ViewSales()

else:

print("Enter correct choice. . . ")

MenuSet()

def runAgain():

runAgn = input("\nwant To Run Again Y/n: ")

while(runAgn.lower() == 'y'):

if(platform.system() == "Windows"):

print(os.system('cls'))

else:

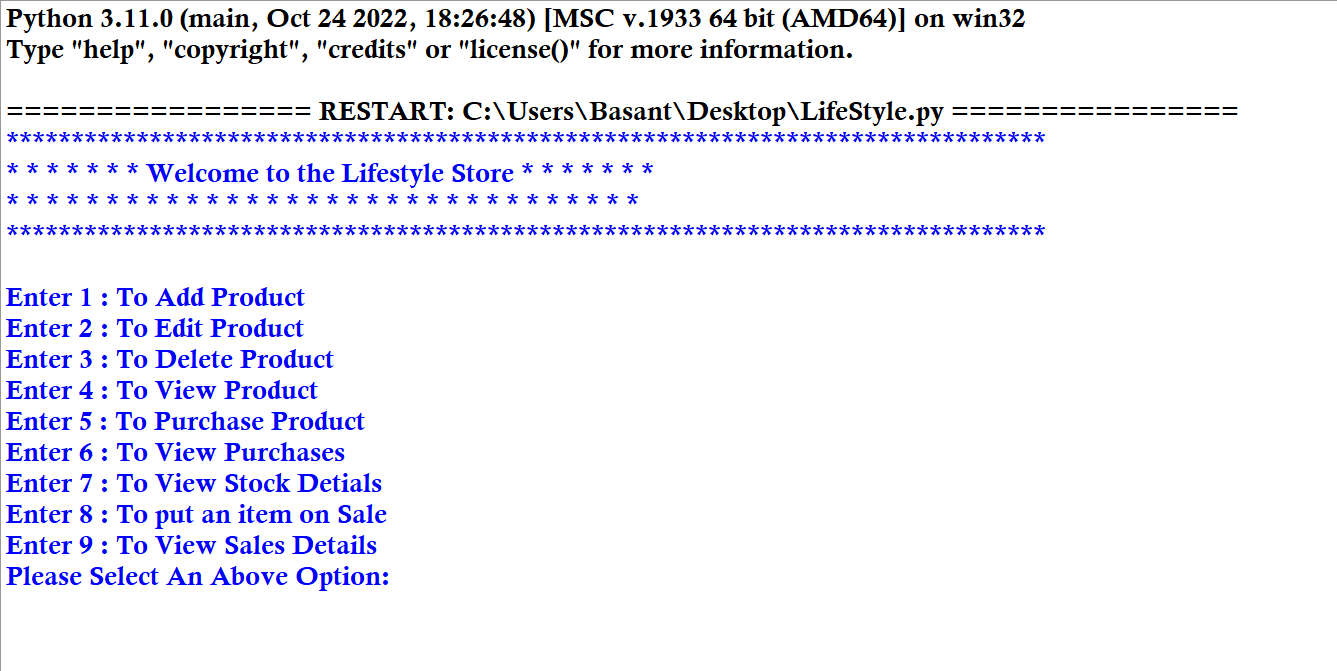
print(os.system('clear'))

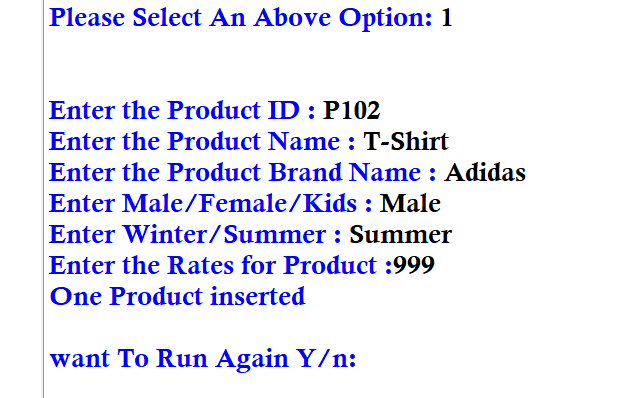
MenuSet()

runAgn = input("\nwant To Run Again Y/n: ")

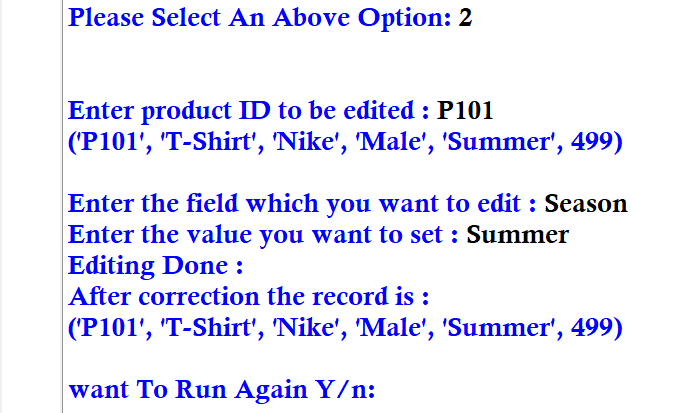
runAgain()

Output:

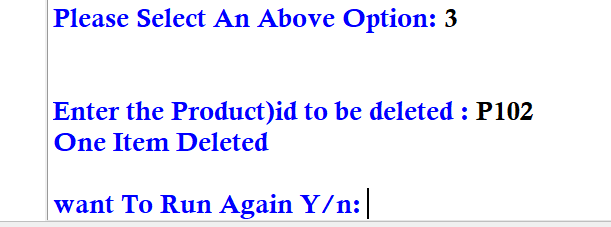
**Main menu:

Add product: **

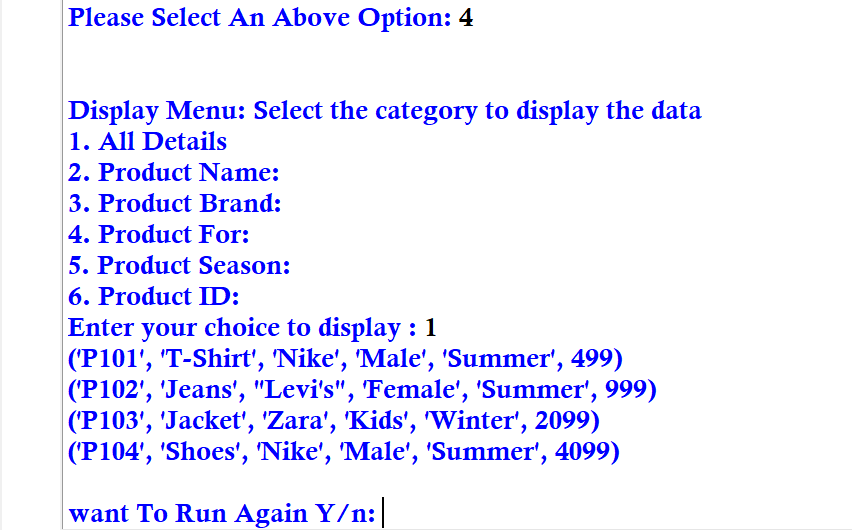
Edit product:

**

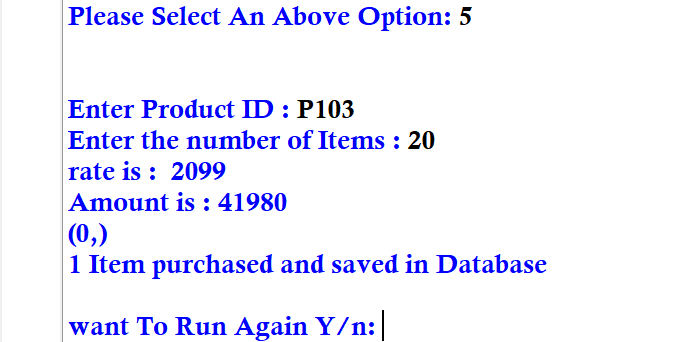
Delete product:

**

View Product:

**

Purchase product:

**

View purchase:

*Graphical user interface, text, application

Description automatically generated*

View stock details:

*Text

Description automatically generated*

Sale item:

*Text

Description automatically generated*

View sales details:

*A picture containing graphical user interface

Description automatically generated*

***Tables created :***

**Product Table-

Purchase Table-

*A picture containing timeline

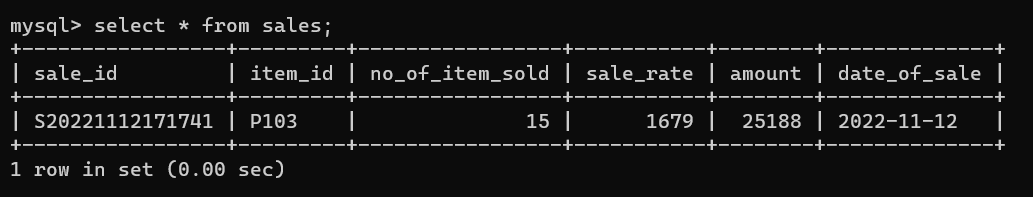
Description automatically generated*

Stock Table-

*A picture containing text

Description automatically generated*

Sales Table-

**

**Future Enhancements**

1. The process of gathering information, diagnosing the problems, then interpreting facts is known as System analysis. It includes recommending system improvements needed, based on the same data.
2. The system is observed as a whole; the inputs need to be identified firstly before tuning them and then the system is subjected to study as a whole to identify the problem areas.
3. Although tuning any system as a whole is a complex procedure, tuning individual statements is not the best as something that is correct for one input may hurt another input's performance.
4. The solutions are given as a proposal. The suggestion is revised on user request and optimal changes are made. This loop terminates as soon as the user is gratified with the proposal.
5. So on the whole, system analysis is done to improve the system performance by monitoring it and obtaining the best throughput possible from it.
6. It would provide more and more details of products which the consumer or the user wants.

**Bibliography**

1. [*MySQL :: MySQL Workbench Manual*](https://dev.mysql.com/doc/workbench/en/)
2. [*https://www.mysql.com/support/supportedplatforms/workbench.html*](https://www.mysql.com/support/supportedplatforms/workbench.html)*.*
3. [*MySQL :: MySQL Connector/Python Release Notes :: Changes in MySQL Connector/Python 8.0.27 (2021-10-19, General Availability)*](https://dev.mysql.com/doc/relnotes/connector-python/en/news-8-0-27.html)
4. [*MySQL :: MySQL Connector/Python Developer Guide*](https://dev.mysql.com/doc/connector-python/en/)
5. [*https://dev.mysql.com/doc/connector-python/en/connector-python-installation.html*](https://dev.mysql.com/doc/connector-python/en/connector-python-installation.html)
6. [*MySQL :: MySQL Connector/Python Developer Guide :: 2 Guidelines for Python Developers*](https://dev.mysql.com/doc/connector-python/en/connector-python-coding.html)
7. [*MySQL :: MySQL Connector/Python Developer Guide :: 4 Connector/Python Installation*](https://dev.mysql.com/doc/connector-python/en/connector-python-installation.html)
8. [*MySQL :: MySQL for Visual Studio*](https://dev.mysql.com/doc/visual-studio/en/)
9. [*MySQL :: MySQL for Visual Studio :: 4 Making a Connection*](https://dev.mysql.com/doc/visual-studio/en/visual-studio-making-a-connection.html)
10. [*MySQL :: MySQL Connector/Python Release Notes :: Changes in MySQL Connector/Python 8.0.27 (2021-10-19, General Availability)*](https://dev.mysql.com/doc/relnotes/connector-python/en/news-8-0-27.html)
11. [*https://www.scribd.com*](https://www.scribd.com)
12. [*https://www.slideshare.net*](https://www.slideshare.net)
13. [*https://en.wikipedia.org*](https://en.wikipedia.org)
14. [*https://www.geeksforgeeks.org/*](https://www.geeksforgeeks.org/)