**Course: Introduction to Database Systems**

**Project Report**

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr no.** | **Topic** | **Page no.** |
| **1** | Storyline |  |
| **2** | Components of Database Design |  |
| **3** | Entity Relationship Diagram |  |
| **4** | Relational Model |  |
| **5** | Normalization |  |
| **6** | SQL Queries |  |

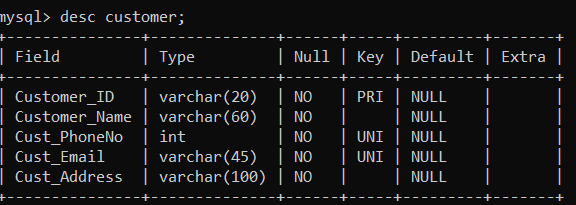
**I. Storyline**

This Project is based on a Jewelry Inventory & Billing System, that can be used by the store employees for easy and seamless management of data such as products, stock, customers, suppliers, employees, sales, payments etc.

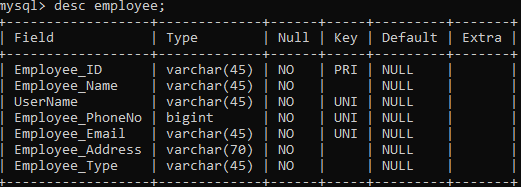
* The Product table is used to store the data of the product, the inventory table stores the information about the product quantity and which supplier supply’s the product.
* The Customer table stores Information about the customers, the employee table as well as the supplier table do the same.
* When a customer purchases a product data is input into the payments and the purchase table at the same time which holds information about the sale of a product and the payment.
* All these functions are being performed by a python code which makes the whole system more efficient.
* A python library called Report Lab is being used to auto generate a pdf invoice when a customer purchases a product.
* A .TXT file is also generated which contains the details of the monthly sales and it can be read using the python code.
* Since the price of gold and silver fluctuate everyday a python id defined which calculates the jewelry price and updates the whole product table in the database with a changed price.

**II. Components of Database Design**

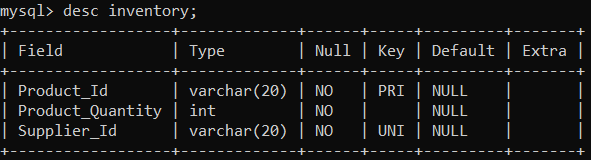
**1.Customer**



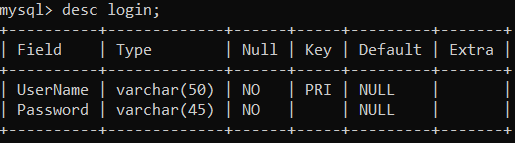
**2. Employee**

****

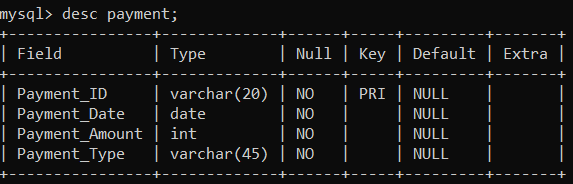
**3.Inventory**



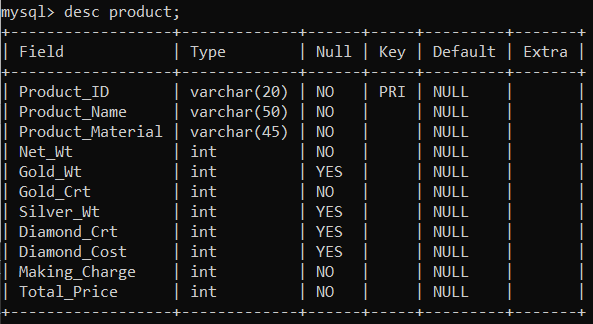
**4. Login**



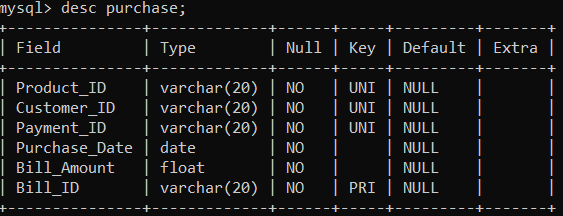
**5. Payment**



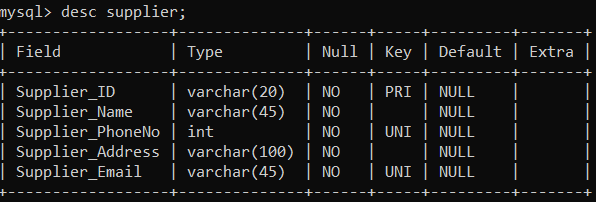
**6. Product**



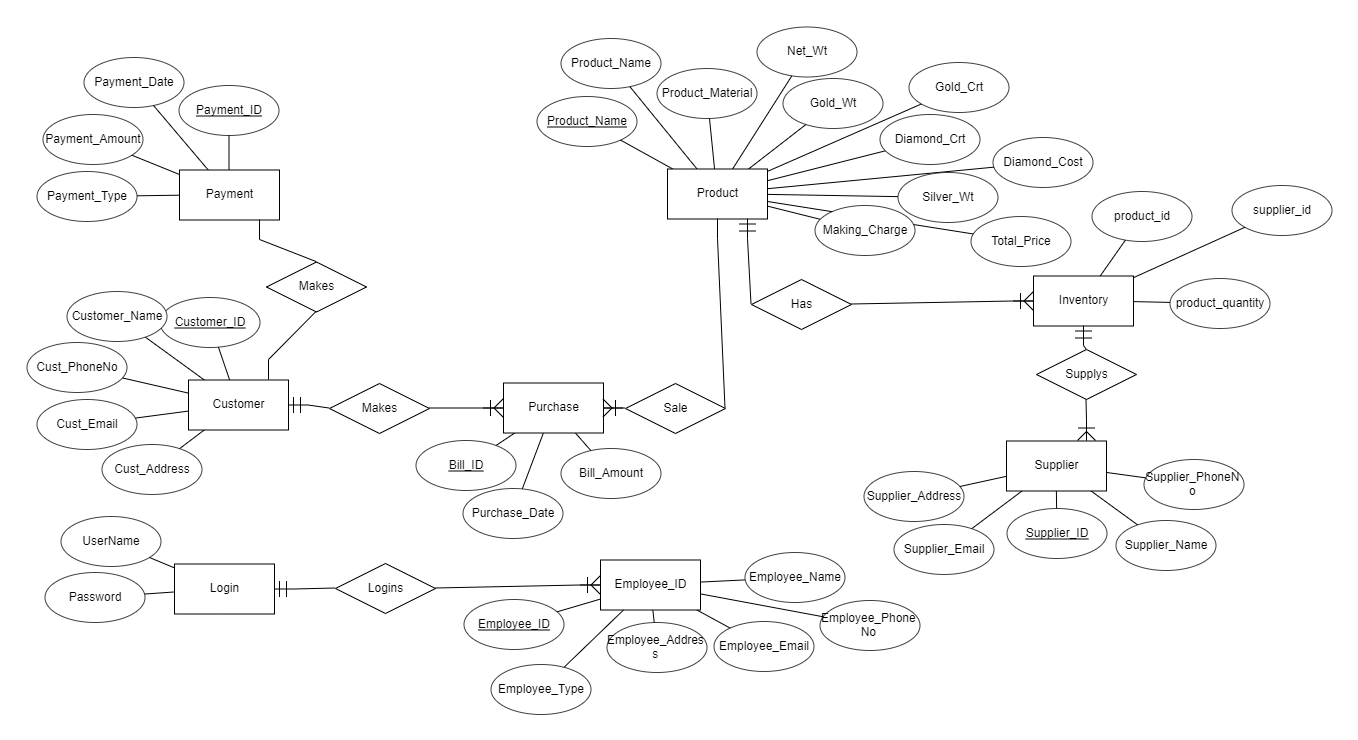
**7. Purchase**



**8. Supplier**



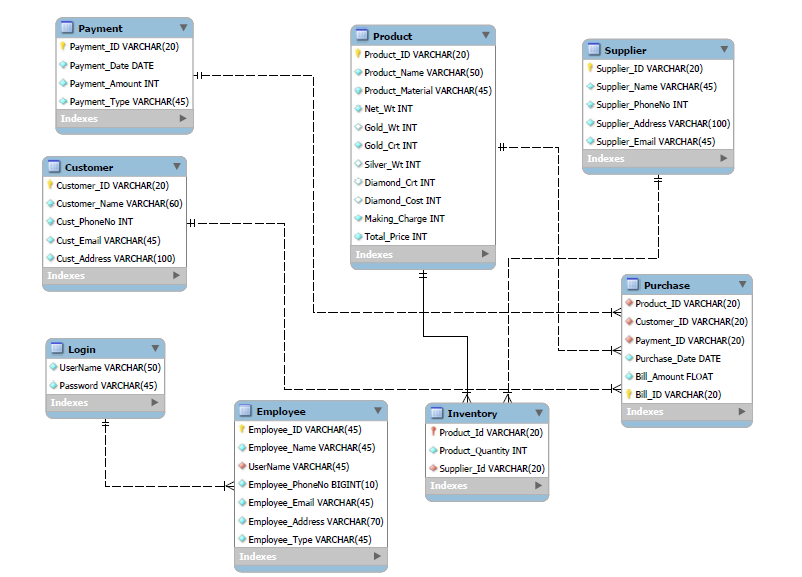
**III. Entity Relationship Diagram**

****

**Relational Schema**

* Customer(customer\_id,customer\_name,cust\_phoneno,cust\_email,cust\_address)
* Employee(employee\_id,employee\_name,username\*,employee\_phoneno,employee\_email,employee\_address,employee\_type)
* Inventory(product\_id\*, product\_quantity, supplier\_id\*)
* Login(username,password)
* Product(product\_id,product\_name,product\_material,net\_wt,gold\_wt,gold\_crt,siler\_wt,diamond\_crt,diamond\_cost,making\_charge,total\_price)
* Purchase(bill\_id,product\_id\*,customer\_id\*,payment\_id\*,purchase\_date,bill\_amount)
* Supplier(supplier\_id,supplier\_name,supplier\_phoneno,supplier\_address,supplier\_email)
* Payment(payment\_id, payment\_date, payment\_amount, payment\_type)

**IV. Relational Model**



**V. Normalization**

**1.For 1 NF:**

• No multivalued attribute, only single valued

• All attributes are dependent on primary key

So, it is in 1NF Form

**2. For 2 NF:**

• It is in 1 NF

• There are no partial dependencies

So, it is in 2 NF form

**3. For 3 NF:**

• It is in 2 NF

• There are no transitive dependencies

So, it is in 3 NF form

**VI. SQL Queries**

Using a DBMS software (SQLite3 or MySQL or any other of your choice):

* Create the tables

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION';

-- -----------------------------------------------------

-- Schema JIBS

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema JIBS

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `JIBS` DEFAULT CHARACTER SET utf8 ;

USE `JIBS` ;

-- -----------------------------------------------------

-- Table `JIBS`.`Product`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Product` (

`Product\_ID` VARCHAR(20) NOT NULL,

`Product\_Name` VARCHAR(50) NOT NULL,

`Product\_Material` VARCHAR(45) NOT NULL,

`Net\_Wt` INT NOT NULL,

`Gold\_Wt` INT NULL,

`Gold\_Crt` INT NOT NULL,

`Silver\_Wt` INT NULL,

`Diamond\_Crt` INT NULL,

`Diamond\_Cost` INT NULL,

`Making\_Charge` INT NOT NULL,

`Total\_Price` INT NOT NULL,

PRIMARY KEY (`Product\_ID`),

UNIQUE INDEX `Product\_ID\_UNIQUE` (`Product\_ID` ASC) VISIBLE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `JIBS`.`Customer`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Customer` (

`Customer\_ID` VARCHAR(20) NOT NULL,

`Customer\_Name` VARCHAR(60) NOT NULL,

`Cust\_PhoneNo` INT NOT NULL,

`Cust\_Email` VARCHAR(45) NOT NULL,

`Cust\_Address` VARCHAR(100) NOT NULL,

PRIMARY KEY (`Customer\_ID`),

UNIQUE INDEX `Customer\_ID\_UNIQUE` (`Customer\_ID` ASC) VISIBLE,

UNIQUE INDEX `Cust\_PhoneNo\_UNIQUE` (`Cust\_PhoneNo` ASC) VISIBLE,

UNIQUE INDEX `Cust\_Email\_UNIQUE` (`Cust\_Email` ASC) VISIBLE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `JIBS`.`Payment`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Payment` (

`Payment\_ID` VARCHAR(20) NOT NULL,

`Payment\_Date` DATE NOT NULL,

`Payment\_Amount` INT NOT NULL,

`Payment\_Type` VARCHAR(45) NOT NULL,

PRIMARY KEY (`Payment\_ID`),

UNIQUE INDEX `Payment\_ID\_UNIQUE` (`Payment\_ID` ASC) VISIBLE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `JIBS`.`Supplier`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Supplier` (

`Supplier\_ID` VARCHAR(20) NOT NULL,

`Supplier\_Name` VARCHAR(45) NOT NULL,

`Supplier\_PhoneNo` INT NOT NULL,

`Supplier\_Address` VARCHAR(100) NOT NULL,

`Supplier\_Email` VARCHAR(45) NOT NULL,

UNIQUE INDEX `Supplier\_ID\_UNIQUE` (`Supplier\_ID` ASC) VISIBLE,

PRIMARY KEY (`Supplier\_ID`),

UNIQUE INDEX `Supplier\_PhoneNo\_UNIQUE` (`Supplier\_PhoneNo` ASC) VISIBLE,

UNIQUE INDEX `Supplier\_Email\_UNIQUE` (`Supplier\_Email` ASC) VISIBLE)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `JIBS`.`Login`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Login` (

`UserName` VARCHAR(50) NOT NULL,

`Password` VARCHAR(45) NOT NULL,

UNIQUE INDEX `UserName\_UNIQUE` (`UserName` ASC))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `JIBS`.`Purchase`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Purchase` (

`Product\_ID` VARCHAR(20) NOT NULL,

`Customer\_ID` VARCHAR(20) NOT NULL,

`Payment\_ID` VARCHAR(20) NOT NULL,

`Purchase\_Date` DATE NOT NULL,

`Bill\_Amount` FLOAT NOT NULL,

`Bill\_ID` VARCHAR(20) NOT NULL,

PRIMARY KEY (`Bill\_ID`),

INDEX `proid\_idx` (`Product\_ID` ASC) VISIBLE,

INDEX `payid\_idx` (`Payment\_ID` ASC) VISIBLE,

INDEX `custid\_idx` (`Customer\_ID` ASC) VISIBLE,

UNIQUE INDEX `Bill\_ID\_UNIQUE` (`Bill\_ID` ASC) VISIBLE,

UNIQUE INDEX `Product\_ID\_UNIQUE` (`Product\_ID` ASC) VISIBLE,

UNIQUE INDEX `Customer\_ID\_UNIQUE` (`Customer\_ID` ASC) VISIBLE,

UNIQUE INDEX `Payment\_ID\_UNIQUE` (`Payment\_ID` ASC) VISIBLE,

CONSTRAINT `proidx`

FOREIGN KEY (`Product\_ID`)

REFERENCES `JIBS`.`Product` (`Product\_ID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `payid`

FOREIGN KEY (`Payment\_ID`)

REFERENCES `JIBS`.`Payment` (`Payment\_ID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `custid`

FOREIGN KEY (`Customer\_ID`)

REFERENCES `JIBS`.`Customer` (`Customer\_ID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `JIBS`.`Employee`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Employee` (

`Employee\_ID` VARCHAR(45) NOT NULL,

`Employee\_Name` VARCHAR(45) NOT NULL,

`UserName` VARCHAR(45) NOT NULL,

`Employee\_PhoneNo` BIGINT(10) NOT NULL,

`Employee\_Email` VARCHAR(45) NOT NULL,

`Employee\_Address` VARCHAR(70) NOT NULL,

`Employee\_Type` VARCHAR(45) NOT NULL,

PRIMARY KEY (`Employee\_ID`),

INDEX `userlog\_idx` (`UserName` ASC) VISIBLE,

UNIQUE INDEX `UserName\_UNIQUE` (`UserName` ASC) VISIBLE,

UNIQUE INDEX `Employee\_ID\_UNIQUE` (`Employee\_ID` ASC) VISIBLE,

UNIQUE INDEX `Employee\_Email\_UNIQUE` (`Employee\_Email` ASC) VISIBLE,

UNIQUE INDEX `Employee\_PhoneNo\_UNIQUE` (`Employee\_PhoneNo` ASC) VISIBLE,

CONSTRAINT `userlog`

FOREIGN KEY (`UserName`)

REFERENCES `JIBS`.`Login` (`UserName`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `JIBS`.`Inventory`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `JIBS`.`Inventory` (

`Product\_Id` VARCHAR(20) NOT NULL,

`Product\_Quantity` INT NOT NULL,

`Supplier\_Id` VARCHAR(20) NOT NULL,

PRIMARY KEY (`Product\_Id`),

UNIQUE INDEX `Supplier\_Id\_UNIQUE` (`Supplier\_Id` ASC) VISIBLE,

UNIQUE INDEX `Product\_Id\_UNIQUE` (`Product\_Id` ASC) VISIBLE,

CONSTRAINT `supid`

FOREIGN KEY (`Supplier\_Id`)

REFERENCES `JIBS`.`Supplier` (`Supplier\_ID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `proid`

FOREIGN KEY (`Product\_Id`)

REFERENCES `JIBS`.`Product` (`Product\_ID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

* Populate the tables (insert some meaningful data, at least 10 tuples for each relation)

INSERT INTO PRODUCT VALUES("P001","GOLD BRACELET","GOLD",30,30,22,0,0,0,8000,0);

INSERT INTO PRODUCT VALUES("P002","GOLD CHAIN","GOLD",20,20,22,0,0,0,5000,0);

INSERT INTO PRODUCT VALUES("P003","GOLD RING","GOLD",10,10,18,0,0,0,2500,0);

INSERT INTO PRODUCT VALUES("P004","GOLD EARRING","GOLD",15,15,18,0,0,0,2800,0);

INSERT INTO PRODUCT VALUES("P005","DIAMOND CHAIN","DIAMOND",10,0,0,0,1,50000,10000,0);

INSERT INTO PRODUCT VALUES("P006","DIAMOND & SILVER RING","DIAMOND & SILVER",25,0,0,10,1,30000,12000,0);

INSERT INTO PRODUCT VALUES("P007","SILVER CHAIN","SILVER",35,0,0,35,0,0,2500,0);

INSERT INTO PRODUCT VALUES("P008","SILVER RING","GOLD",20,0,0,20,0,0,2000,0);

INSERT INTO PRODUCT VALUES("P009","GOLD AND SILVER BRACELET","GOLD & SILVER",30,15,22,15,0,0,10000,0);

INSERT INTO PRODUCT VALUES("P010","SILVER EARRING","SILVER",20,0,0,20,0,0,2500,0);

INSERT INTO INVENTORY VALUES("P001",5,"S001");

INSERT INTO INVENTORY VALUES("P002",4,"S002");

INSERT INTO INVENTORY VALUES("P003",5,"S002");

INSERT INTO INVENTORY VALUES("P004",9,"S003");

INSERT INTO INVENTORY VALUES("P005",8,"S001");

INSERT INTO INVENTORY VALUES("P006",15,"S005");

INSERT INTO INVENTORY VALUES("P007",11,"S004");

INSERT INTO INVENTORY VALUES("P008",18,"S004");

INSERT INTO INVENTORY VALUES("P009",25,"S005");

INSERT INTO INVENTORY VALUES("P010",25,"S003");

INSERT INTO SUPPLIER VALUES("S001","ZAR JEWELS",8657499165,"Mathuradas mill compound,N. M. Joshi Marg, Lower Parel.W","info@zarjewels.com");

INSERT INTO SUPPLIER VALUES("S002","WAMAN HARI PETHE JEWELERS",2228335833,"Jeevan Mandir, Shimpoli Road Boriwali (w, opp. Gokhale High School, Mumbai, Maharashtra 400092","info@whp.com");

INSERT INTO SUPPLIER VALUES("S003","CARATLANE",9920222500,"Infiniti Mall 2, GF 041, Malad West, Mumbai 400064","info@caratlane.com");

INSERT INTO SUPPLIER VALUES("S004","TANISHQ",7738892988,"Vini Eligence Building, Ground Floor, L.T.Road, Borivali West, Mumbai, Maharashtra 400092","info@tanishq.com");

INSERT INTO SUPPLIER VALUES("S005","OMKAR JEWELS",8976589344,"34 Vidyarthi Makret, Ground Floor,Govind Nagar, Kanpur","info@omkar.com");

INSERT INTO CUSTOMER VALUES("C001","UTKARSH KHANNA",9871163635,"UKHANNA2003@GMAIL.COM","43,MAHARAJA ASHOK,JUHU");

INSERT INTO CUSTOMER VALUES("C002","PRANAV DHADWAL",9594094930,"PRANAV.DHADWAL@GMAIL.COM","55 Pali Naka Bandra , Mumbai,Mumbai,400051,India");

INSERT INTO CUSTOMER VALUES("C003","ARYAN PENTA",8454824602,"ARYAN.PENTA@GMAIL.COM","161 Jain Laxmi Mansion D Bhadkamkar Rd Opp Navjeevan Soc, Mumbai,Mumbai,400008,India");

INSERT INTO CUSTOMER VALUES("C004","NEHAAL CHOUDHARY",9872264645,"NEHAAL@GMAIL.COM","44,CHAND SOCIETY,JUHU");

INSERT INTO CUSTOMER VALUES("C005","ABBHIRUP BUDHIRAJA",9819623137,"ABBHIRUP@GMAIL.COM","Goregaon , Mumbai,Mumbai,400063,India");

INSERT INTO CUSTOMER VALUES("C006","ROHIN PITHWA",9867169420,"ROHINPITHWA@GMAIL.COM","114 -a Sajid Tower S V Rd Andheri , Mumbai,Mumbai,400058,India");

INSERT INTO CUSTOMER VALUES("C007","FRANIA CHETTIAR",9167529432,"CHETTIARFRANIA@GMAIL.COM","29 So- Lucky Corner Bldg Andheri Kurla Road Mumbai,400093,India");

INSERT INTO CUSTOMER VALUES("C008","DURVESH GADE",8451977187,"DURVESHGADE@GMAIL.COM","85 E/, Delhi,Mumbai,110019,India");

INSERT INTO CUSTOMER VALUES("C009","CHETAN YADAV",7977065724,"CHETAN.Y3@GMAIL.COM","74 /i Islam Bldg V N Road Crawford Market, Mumbai,Mumbai,400001,India");

INSERT INTO EMPLOYEE VALUES("E005","ADVIK BASIA","ADVIKB01",9782264645,"ADVIK.B@GMAIL.COM","D502,VISTAS,POWAI","SALES PERSON");

INSERT INTO EMPLOYEE VALUES("E002","ESHAN TRIVEDI","ESHANB02",9877775645,"ESHAN.009@GMAIL.COM","D2,PEDER,GRANT ROAD","SALES PERSON");

INSERT INTO EMPLOYEE VALUES("E003","JUSTIN PARMAR","JUSTINB03",9767389200,"JUSTIN98@GMAIL.COM","400,DELTA,JUHU","SALES PERSON");

INSERT INTO EMPLOYEE VALUES("E004","ADITI JAIN","ADITIB04",96452313132,"ADITI90@GMAIL.COM","A02,VISTARA,ANDHERI","SALES PERSON");

INSERT INTO EMPLOYEE VALUES("E001","UTKARSH KHANNA","UK22",9871163635,"UTKARSH.K@GMAIL.COM","44,ASHOK APARTMENT JUHU","MANAGER");

INSERT INTO LOGIN VALUES("ESHAN02","TRIVEDI");

INSERT INTO LOGIN VALUES("JUSTIN03","PARMAR");

INSERT INTO LOGIN VALUES("ADITI04","JAIN");

INSERT INTO LOGIN VALUES(“UK22”,”UTKARSH”)

INSERT INTO LOGIN VALUES(“ADVIKB01”,”ADVIK123”)

* Run SQL queries (minimum 15) covering **all concepts** learned in the class
* import mysql.connector  
  from reportlab.platypus import SimpleDocTemplate, Table, Paragraph, TableStyle  
  from reportlab.pdfgen import canvas  
  from reportlab.lib import colors  
    
  from reportlab.lib.pagesizes import A4, landscape  
    
  from reportlab.lib.styles import getSampleStyleSheet  
    
    
  # function to accquire DB connectivity  
  def getDbConnection():  
   conn = mysql.connector.connect(host='localhost', user='root', password='root', database='JIBS')  
   if conn.is\_connected():  
   # print('Connection Sucessfully Established With Database !')  
   # print()  
   return conn  
   else:  
   print('Connection Not Accquired !')  
   return 0  
    
    
  def NorMenu():  
   print(' ----------------------------------------------------------- ')  
   print(" | LOGGED IN AS SALES PERSON | ")  
   print(' |-----------------------------------------------------------| ')  
   print(' | JEWELLERY INVENTORY AND BILLING SYSTEM |')  
   print(' |-----------------------------------------------------------|')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* M E N U \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print(' \*\* \*\*')  
   print(' \*\* 1. Show List Of Products \*\*')  
   print(' \*\* 2. Search A Product \*\*')  
   print(' \*\* 3. Add A Customer \*\*')  
   print(' \*\* 4. Search Customer \*\*')  
   print(' \*\* 5. Purchase \*\*')  
   print(' \*\* 6. Exit JIBS \*\*')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print()  
    
    
  def MangMenu():  
   print(' ----------------------------------------------------------- ')  
   print(" | LOGGED IN AS MANAGER | ")  
   print(' |-----------------------------------------------------------| ')  
   print(' | JEWELLERY INVENTORY AND BILLING SYSTEM |')  
   print(' |-----------------------------------------------------------|')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* M E N U \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print(' \*\* \*\*')  
   print(' \*\* 1. Add A Product \*\*')  
   print(' \*\* 2. Add A Supplier \*\*')  
   print(' \*\* 3. Show List Of Products \*\*')  
   print(' \*\* 4. Search A Product \*\*')  
   print(' \*\* 5. Add A Customer \*\*')  
   print(' \*\* 6. Search Customer \*\*')  
   print(' \*\* 7. Purchase \*\*')  
   print(' \*\* 8. Update Product Quantity \*\*')  
   print(' \*\* 9. Monthly Sales Report \*\*')  
   print(' \*\* 10. Add New Employee \*\*')  
   print(' \*\* 11. Exit JIBS \*\*')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print(' \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*')  
   print()  
    
    
  def login():  
   isValidUser = 0  
   while (isValidUser != 1 and isValidUser != 2):  
   print()  
   print(' ------------------------ ')  
   print(' | LOGIN PAGE | ')  
   print(' ------------------------ ')  
   print()  
   uname = input(' Enter Your Username : ')  
   password = input(' Enter Your Password : ')  
   print()  
   isValidUser = authenticate(uname, password)  
   return isValidUser  
    
    
  def authenticate(uname, password):  
   conn = getDbConnection()  
   mycur = conn.cursor()  
   query = 'select l.username,l.password,e.employee\_type from login as l,employee as e where l.username=e.username and l.USERNAME=%s AND l.PASSWORD=%s;'  
   data = (uname, password)  
   mycur.execute(query, data)  
   result = mycur.fetchall()  
   validLogin = 0  
   conn.commit()  
   conn.close()  
   for i in result:  
   if (i[0] == uname and i[1] == password and i[2] == "MANAGER"):  
   print()  
   print(' -> Login Sucessful !')  
   print()  
   validLogin = 1  
    
   elif (i[0] == uname and i[1] == password and i[2] == "SALES PERSON"):  
   print()  
   print(' -> Login Sucessful !')  
   print()  
   validLogin = 2  
    
   if validLogin == 0:  
   print('Invalid Username Or Password,Please Try Again ! ')  
    
   return validLogin  
    
    
    
    
  def addProduct():  
   conn = getDbConnection()  
   suppId = input("Enter The Supplier Id : ")  
   productName = input(' Enter The Product Name : ')  
   material = input(' Enter The Product Material : ')  
   productWt = int(input(' Enter The Product Weight : '))  
   proQty = int(input(' Enter The Product Quantity : '))  
   goldWt = int(input(' Enter The Gold Weight(Gm) : '))  
   goldCrt = int(input(' Enter The Gold Carat : '))  
   silverWt = int(input(' Enter The Silver Weight(Gm) : '))  
   diamondCrt = int(input(' Enter The Diamond Carat : '))  
   diamondCost = int(input(' Enter The Diamond Cost : '))  
   makeCharge = int(input(' Enter The Making Charges : '))  
   productPrice = 0  
   proStr = "P0"  
   mycur1 = conn.cursor()  
   mycur2 = conn.cursor()  
   mycur3 = conn.cursor()  
   queryProId = "SELECT COUNT(PRODUCT\_ID) FROM PRODUCT"  
   mycur3.execute(queryProId)  
   pid = mycur3.fetchone()  
   autoProId = pid[0] + 1  
   productId = proStr + str(autoProId)  
   query1 = '''INSERT INTO PRODUCT(Product\_ID,Product\_Name,Product\_Material,NET\_WT,Gold\_Wt,Gold\_Crt,Silver\_Wt,Diamond\_Crt,Diamond\_Cost,Making\_Charge,Total\_Price)VALUES(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s);'''  
   value1 = (  
   productId, productName, material, productWt, goldWt, goldCrt, silverWt, diamondCrt, diamondCost, makeCharge,  
   productPrice)  
   mycur1.execute(query1, value1)  
   query2 = 'INSERT INTO INVENTORY(PRODUCT\_ID,PRODUCT\_QUANTITY,SUPPLIER\_ID)VALUES(%s,%s,%s)'  
   value2 = (productId, proQty, suppId)  
   mycur2.execute(query2, value2)  
   print()  
   print(' ->Record Inserted !')  
   print()  
   conn.commit()  
   conn.close()  
   CalGoldRate()  
    
    
  def CalGoldRate():  
   GoldRate = int(input("Enter Today's Gold Rate : "))  
   SilverRate = int(input("Enter Today's Silver Rate : "))  
   conn = getDbConnection()  
   queryCal = 'select Product\_ID,Net\_Wt,Gold\_Wt,Diamond\_Cost,Silver\_Wt,Making\_Charge,Total\_Price from PRODUCT ORDER BY PRODUCT\_ID;'  
   mycur = conn.cursor()  
   mycur.execute(queryCal)  
   result = mycur.fetchall()  
   for i in result:  
   ProId = i[0]  
   NetWt = i[1]  
   Gold\_Wt = i[2]  
   Diamond\_Cost = i[3]  
   Silver\_Wt = i[4]  
   Making\_Charge = i[5]  
   Total\_Price = i[6]  
   calGold = GoldRate \* Gold\_Wt  
   calSilver = SilverRate \* Silver\_Wt  
   totalCost = calGold + calSilver + Diamond\_Cost + Making\_Charge  
   queryUp = 'UPDATE PRODUCT SET TOTAL\_PRICE = %s WHERE PRODUCT\_ID=%s'  
   x = (totalCost, ProId)  
   mycur.execute(queryUp, x)  
   conn.commit()  
   conn.close()  
    
    
  def addCustomer():  
   conn = getDbConnection()  
   customerName = input(' Enter The Customer Name : ')  
   custPh = int(input(' Enter The Customer Phone Number : '))  
   custEmail = input(' Enter The Customer Email ID : ')  
   custAddress = input(' Enter The Customer Address : ')  
   custStr = "C0"  
   mycur = conn.cursor()  
   mycurCust = conn.cursor()  
   queryCustomerId = "SELECT COUNT(CUSTOMER\_ID) FROM CUSTOMER"  
   mycurCust.execute(queryCustomerId)  
   cid = mycurCust.fetchone()  
   custId = cid[0] + 1  
   customerId = custStr + str(custId)  
   queryCust = '''INSERT INTO CUSTOMER(Customer\_ID,Customer\_Name,Cust\_PHONENO,Cust\_Email,Cust\_Address)VALUES(%s,%s,%s,%s,%s);'''  
   valueCust = (customerId, customerName, custPh, custEmail, custAddress)  
   mycur.execute(queryCust, valueCust)  
   print()  
   print(' ->Record Inserted !')  
   print()  
   conn.commit()  
   conn.close()  
    
    
  def listProduct():  
   conn = getDbConnection()  
   query3 = 'select P.PRODUCT\_ID,P.PRODUCT\_NAME,P.PRODUCT\_MATERIAL,P.NET\_WT,P.GOLD\_CRT,P.DIAMOND\_CRT,I.PRODUCT\_QUANTITY,P.TOTAL\_PRICE from PRODUCT AS P,INVENTORY AS I WHERE I.PRODUCT\_ID=P.PRODUCT\_ID ORDER BY P.PRODUCT\_ID;'  
   mycur = conn.cursor()  
   mycur.execute(query3)  
   result = mycur.fetchall()  
   print()  
   print('-' \* 200)  
   print('-' \* 200)  
   print(  
   " {0:20}{1:30}{2:30}{3:10}{4:10}{5:10}{6:10}{7:20}".format(' ID ', ' NAME ', ' MATERIAL ', ' NET\_WT ',  
   ' GOLD\_CRT ', ' DIAMOND\_CRT ', ' QTY ',  
   ' COST '))  
   print('-' \* 200)  
   print('-' \* 200)  
   for x in result:  
   print(" {0:20}{1:30}{2:30}{3:10}{4:10}{5:10}{6:10}{7:20}".format(x[0], x[1], x[2], x[3], x[4], x[5], x[6],  
   x[7]))  
   print('-' \* 200)  
   print('-' \* 200)  
   print()  
   print()  
   conn.commit()  
   conn.close()  
    
    
  def searchProduct():  
   op = 0  
   while op >= 0 < 4:  
   print(' ------------------------------- ')  
   print(' | 1. Search Product By ID |')  
   print(' |-------------------------------|')  
   print(' | 2. Search Product By Name |')  
   print(' |-------------------------------|')  
   print(' | 3. Search Product By Material |')  
   print(' |-------------------------------|')  
   print(' | 4. Go To Main Menu |')  
   print(' ------------------------------- ')  
   print()  
   op = int(input(' Choose An Option From The Menu : '))  
   print()  
    
   if op == 1:  
   conn = getDbConnection()  
   mycur = conn.cursor()  
   productID = input(' Enter The ID Of The Product You Want To Search : ')  
   query6 = 'SELECT \* FROM PRODUCT WHERE PRODUCT\_ID LIKE %s'  
   value6 = ['%' + productID + '%']  
   mycur.execute(query6, value6)  
   x = mycur.fetchall()  
   print()  
   for i in x:  
   print(i)  
   conn.commit()  
   conn.close()  
    
   elif op == 2:  
    
   conn = getDbConnection()  
   mycur = conn.cursor()  
   productName = input(' Enter The Name Of The Product You Want To Search : ')  
   query7 = 'SELECT \* FROM PRODUCT WHERE PRODUCT\_NAME LIKE %s'  
   value7 = ['%' + productName + '%']  
   mycur.execute(query7, value7)  
   x = mycur.fetchall()  
   print()  
   for i in x:  
   print(i)  
   conn.commit()  
   conn.close()  
   elif op == 3:  
    
   conn = getDbConnection()  
   mycur = conn.cursor()  
   material = input(' Enter The Material Of The Product You Want To Search : ')  
   query8 = 'SELECT \* FROM PRODUCT WHERE PRODUCT\_MATERIAL LIKE %s'  
   value8 = [material]  
   mycur.execute(query8, value8)  
   x = mycur.fetchall()  
   print()  
   for i in x:  
   print(i)  
   conn.commit()  
   conn.close()  
   else:  
   break  
    
   if op == 4:  
   return  
    
    
  def searchCustomer():  
   op = 0  
   while op >= 0 < 4:  
   print(' ------------------------------- ')  
   print(' | 1. Search Customer By Name |')  
   print(' |-------------------------------|')  
   print(' | 2. Search Customer By PhoneNo |')  
   print(' |-------------------------------|')  
   print(' | 3. Go To Main Menu |')  
   print(' ------------------------------- ')  
   print()  
   op = int(input(' Choose An Option From The Menu : '))  
   print()  
    
   if op == 1:  
   conn = getDbConnection()  
   mycur = conn.cursor()  
   custName = input(' Enter The Name Of The Customer You Want To Search : ')  
   query6 = 'SELECT \* FROM CUSTOMER WHERE CUSTOMER\_NAME LIKE %s'  
   value6 = ['%' + custName + '%']  
   mycur.execute(query6, value6)  
   x = mycur.fetchall()  
   print()  
   for i in x:  
   print(i)  
   conn.commit()  
   conn.close()  
    
   if op == 2:  
   conn = getDbConnection()  
   mycur = conn.cursor()  
   custPhone = input(' Enter The Phone Number Of The Customer You Want To Search : ')  
   query7 = 'SELECT \* FROM CUSTOMER WHERE CUST\_PHONENO LIKE %s'  
   value7 = ['%' + custPhone + '%']  
   mycur.execute(query7, value7)  
   x = mycur.fetchall()  
   print()  
   for i in x:  
   print(i)  
   conn.commit()  
   conn.close()  
    
   if op == 4:  
   return  
   else:  
   return  
    
    
  def searchSupplier():  
   op = 0  
   while op >= 0 < 4:  
   print(' ------------------------------- ')  
   print(' | 1. Search Supplier By Name |')  
   print(' |-------------------------------|')  
   print(' | 2. Search Supplier By PhoneNo |')  
   print(' |-------------------------------|')  
   print(' | 3. Go To Main Menu |')  
   print(' ------------------------------- ')  
   print()  
   op = int(input(' Choose An Option From The Menu : '))  
   print()  
    
   if op == 1:  
   conn = getDbConnection()  
   mycur = conn.cursor()  
   custName = input(' Enter The Name Of The Customer You Want To Search : ')  
   query6 = 'SELECT \* FROM SUPPLIER WHERE SUPPLIER\_NAME LIKE %s'  
   value6 = ['%' + custName + '%']  
   mycur.execute(query6, value6)  
   x = mycur.fetchall()  
   print()  
   for i in x:  
   print(i)  
   conn.commit()  
   conn.close()  
    
   if op == 2:  
   conn = getDbConnection()  
   mycur = conn.cursor()  
   custPhone = input(' Enter The Phone Number Of The Customer You Want To Search : ')  
   query7 = 'SELECT \* FROM SUPPLIER WHERE SUPPLIER\_PHONENO LIKE %s'  
   value7 = ['%' + custPhone + '%']  
   mycur.execute(query7, value7)  
   x = mycur.fetchall()  
   print()  
   for i in x:  
   print(i)  
   conn.commit()  
   conn.close()  
    
   if op == 4:  
   return  
   else:  
   return  
    
    
  def addSupplier():  
   conn = getDbConnection()  
   supplierName = input(" Enter The Supplier's Name : ")  
   supplierPh = int(input(" Enter The Supplier's Phone Number: "))  
   supplierEmail = input(" Enter The Supplier's Email ID : ")  
   supplierAddress = input(" Enter The Supplier's Address : ")  
   mycur = conn.cursor()  
   mycurSupp = conn.cursor()  
   supStr = "S00"  
   querySupplierId = "SELECT COUNT(SUPPLIER\_ID) FROM SUPPLIER"  
   mycurSupp.execute(querySupplierId)  
   sid = mycurSupp.fetchone()  
   supId = sid[0] + 1  
   supplierId = supStr + str(supId)  
   querySupp = '''INSERT INTO SUPPLIER(SUPPLIER\_ID,SUPPLIER\_Name,SUPPLIER\_PHONENO,SUPPLIER\_Address,SUPPLIER\_EMAIL)VALUES(%s,%s,%s,%s,%s);'''  
   valueSupp = (supplierId, supplierName, supplierPh, supplierAddress, supplierEmail)  
   mycur.execute(querySupp, valueSupp)  
   print()  
   print(' ->Record Inserted !')  
   print()  
   conn.commit()  
   conn.close()  
    
    
  def verifyProduct(proId):  
   conn = getDbConnection()  
   mycurB = conn.cursor()  
   queryChkProduct = 'SELECT PRODUCT\_QUANTITY FROM INVENTORY WHERE PRODUCT\_ID=%s ;'  
   valueChkProduct = (proId,)  
   mycurB.execute(queryChkProduct, valueChkProduct)  
   chk1 = mycurB.fetchone()  
   conn.commit()  
   conn.close()  
   return chk1[0]  
    
    
  def purchase():  
   count = 0  
   CustId = input(' Enter Customer ID : ')  
   proId = input(' Enter Product ID : ')  
   payType = input(' Enter Payment Type : ')  
   isProductValid = verifyProduct(proId)  
   payStr = "PAY0"  
   billStr = "B0"  
   if isProductValid > 0:  
   conn = getDbConnection()  
   mycurPay = conn.cursor()  
   mycurPurchase = conn.cursor()  
   mycurSel = conn.cursor()  
   mycurUpd = conn.cursor()  
   mycurAmt = conn.cursor()  
   mycurPayment = conn.cursor()  
   mycurAutoBill = conn.cursor()  
   mycurBill1 = conn.cursor()  
   mycurBill2 = conn.cursor()  
   mycurInv = conn.cursor()  
   mycurCustomer = conn.cursor()  
   queryPayId = "SELECT COUNT(PAYMENT\_ID) FROM PAYMENT"  
   mycurPayment.execute(queryPayId)  
   pid1 = mycurPayment.fetchone()  
   paymentId = pid1[0] + 1  
   PayId = payStr + str(paymentId)  
    
   queryBillId = "SELECT COUNT(BILL\_ID) FROM PURCHASE"  
   mycurAutoBill.execute(queryBillId)  
   bid1 = mycurAutoBill.fetchone()  
   autoBillId = bid1[0] + 1  
   billId = billStr + str(autoBillId)  
    
   queryChkProduct = 'SELECT PRODUCT\_QUANTITY FROM INVENTORY WHERE PRODUCT\_ID=%s ;'  
   valueChkProduct = (proId,)  
   mycurSel.execute(queryChkProduct, valueChkProduct)  
   qty = mycurSel.fetchone()  
   print(qty[0])  
   x = qty[0] - 1  
   # print('verifyBook=',chk1)  
   if qty[0] > 0:  
   qp = 'UPDATE INVENTORY SET PRODUCT\_QUANTITY=%s WHERE PRODUCT\_ID=%s ;'  
   vcp = (x, proId)  
   mycurUpd.execute(qp, vcp)  
   chk1 = mycurUpd.fetchone()  
   amtQuery = 'SELECT TOTAL\_PRICE FROM PRODUCT WHERE PRODUCT\_ID=%s'  
   amtVal = (proId,)  
   mycurAmt.execute(amtQuery, amtVal)  
   payAmt = mycurAmt.fetchone()  
   paymentQuery = '''INSERT INTO PAYMENT(PAYMENT\_ID,PAYMENT\_DATE,PAYMENT\_AMOUNT,PAYMENT\_TYPE)VALUES(%s,NOW(),%s,%s)'''  
   count += 1  
   paymentValue = (PayId, payAmt[0], payType)  
   mycurPay.execute(paymentQuery, paymentValue)  
   print()  
   print(' Payment has been Done for Payment ID', PayId, '!')  
   print()  
   purchaseQuery = '''INSERT INTO PURCHASE(PRODUCT\_ID,CUSTOMER\_ID,PAYMENT\_ID,PURCHASE\_DATE,BILL\_AMOUNT,BILL\_ID)VALUES(%s,%s,%s,NOW(),%s,%s)'''  
   valuePurchase = (proId, CustId, PayId, payAmt[0], billId)  
   mycurPurchase.execute(purchaseQuery, valuePurchase)  
   billQuery1 = 'SELECT PUR.CUSTOMER\_ID,PUR.BILL\_ID,PUR.PAYMENT\_ID,PUR.PURCHASE\_DATE,PUR.BILL\_AMOUNT,PAY.PAYMENT\_TYPE FROM PURCHASE AS PUR,PAYMENT AS PAY WHERE PUR.PAYMENT\_ID = PAY.PAYMENT\_ID AND PAY.PAYMENT\_ID=%s'  
   billVal1 = (PayId,)  
   mycurBill1.execute(billQuery1, billVal1)  
   billFetch1 = mycurBill1.fetchall()  
   billQuery2 = 'SELECT PRODUCT\_ID,PRODUCT\_NAME,NET\_WT,GOLD\_WT,GOLD\_CRT,SILVER\_WT,DIAMOND\_CRT,DIAMOND\_COST,MAKING\_CHARGE FROM PRODUCT WHERE PRODUCT\_ID =%s'  
   billVal2 = (proId,)  
   mycurBill2.execute(billQuery2, billVal2)  
   billFetch2 = mycurBill2.fetchall()  
   invQuery = 'SELECT BILL\_ID FROM PURCHASE WHERE PAYMENT\_ID = %s'  
   valInv = (PayId,)  
   mycurInv.execute(invQuery, valInv)  
   invFetch = mycurInv.fetchone()  
   for bill in billFetch2:  
    
   tableData2 = [["PRODUCT ID","PRODUCT NAME","NET WT","GOLD WT","GOLD CRT", "SILVER WT", "DIAMOND CRT", "DIAMOND COST", "MAKING CHARGE" ],  
   [bill[0], bill[1], bill[2], bill[3], bill[4], bill[5], bill[6],bill[7],bill[8]]  
   ]  
    
   for bill2 in billFetch1:  
   tableData1 = [  
    
   ["CUSTOMER ID", "BILL ID", "PAYMENT ID", "PURCHASE DATE", "PAYMENT TYPE", "TOTAL AMOUNT"],  
   [bill2[0], bill2[1], bill2[2], bill2[3], bill2[5], bill2[4]]  
   ]  
    
   queryCustomer = 'SELECT CUSTOMER\_NAME,CUST\_PHONENO,CUST\_EMAIL,CUST\_ADDRESS FROM CUSTOMER WHERE CUSTOMER\_ID = %s'  
   valCustomer = (CustId,)  
   mycurCustomer.execute(queryCustomer,valCustomer)  
   custValue = mycurCustomer.fetchall()  
   custVal1 = " "  
   custVal2 = " "  
   custVal3 = " "  
   custVal4 = " "  
   b1 = " "  
   b2 = " "  
   b3 = " "  
   b4 = " "  
   b5 = " "  
   b6 = " "  
   for i in custValue:  
   custVal1 = str(i[0])  
   custVal2 = str(i[1])  
   custVal3 = str(i[2])  
   custVal4 = str(i[3])  
    
   # creating a Document structure with A4 size page  
   fname = invFetch[0]  
   docu = SimpleDocTemplate("C:/Users/ukhan/Desktop/'%s'.pdf"%fname, pagesize=(landscape(A4)))  
   styles = getSampleStyleSheet()  
    
   doc\_style = styles["Heading1"]  
   doc\_style2 = styles["Heading2"]  
   doc\_style3 = styles["Heading2"]  
   doc\_style4 = styles["Heading3"]  
   doc\_style5 = styles["Heading4"]  
   doc\_style.alignment = 1  
   doc\_style2.alignment = 2  
   doc\_style3.alignment = 2  
   doc\_style4.alignment = 0  
   doc\_style5.alignment = 1  
   title = Paragraph("MAAHIRA DIAMOND", doc\_style)  
   title2 = Paragraph("INVOICE", doc\_style)  
   inv = Paragraph("Invoice No . ", doc\_style2)  
   invno = Paragraph(invFetch[0], doc\_style3)  
   addl1 = Paragraph("126/8/24, , VIDYARTHI MARKET", doc\_style5)  
   addl2 = Paragraph("GOVIND NAGAR, KANPUR, Uttar Pradesh", doc\_style5)  
   addl3 = Paragraph("Pincode, 208006", doc\_style5)  
   cust1 = Paragraph(custVal1,doc\_style4)  
   cust2 = Paragraph(custVal2,doc\_style4)  
   cust3 = Paragraph(custVal3,doc\_style4)  
   cust4 = Paragraph(custVal4,doc\_style4)  
   bb1= Paragraph(b1,doc\_style4)  
   bb2= Paragraph(b2,doc\_style4)  
   bb3= Paragraph(b3,doc\_style4)  
   bb4=Paragraph(b4,doc\_style4)  
   bb5=Paragraph(b5,doc\_style4)  
   bb6=Paragraph(b6,doc\_style4)  
    
    
    
    
    
    
    
   style = TableStyle([  
   ('BOX', (0, 0), (-1, -1), 0.20, colors.dimgrey),  
   ('FONT', (0, 0), (-1, 0), 'Helvetica-Bold'),  
   ('INNERGRID', (0, 0), (-1, -1), 0.1, colors.black),  
   ('VALIGN', (0, 0), (-1, -1), 'BOTTOM'),  
   ('ALIGN', (0, 0), (-1, -1), 'CENTER'),  
   ('FONTSIZE', (0, 0), (-1, -1), 10),  
    
   ])  
   TableStyle.Align = 'BOTTOM'  
   TableStyle.vAlign = 'BOTTOM'  
   # creates a table object using the Table() to pass the table data and the style object  
    
   table1 = Table(tableData1, style=style)  
   table2 = Table(tableData2, style=style)  
   # finally, we have to build the actual pdf merging all objects together  
    
   docu.build([title, title2, inv, invno, cust1,cust2,cust3,cust4,bb1,bb2,bb3, table2, table1,bb4,bb5,bb6,addl1, addl2, addl3])  
   print("Your Bill ",billId," has Been generated !")  
   conn.commit()  
   conn.close()  
   else:  
   print("Product Out Of Stock !")  
    
    
    
    
    
    
  def addEmployee():  
   conn = getDbConnection()  
   employeeName = input(" Enter The Employee's Name : ")  
   username = input(" Enter The Employee's Username : ")  
   employeePhone = int(input(" Enter The Employee's Phone Number: : "))  
   employeeEmail = input(" Enter The Employee's Email ID : ")  
   employeeAddress = input(" Enter The Employee's Address : ")  
   employeeType = input(" Enter The Employee Type : ")  
   password = input(" Enter The Employee's Login Password : ")  
   mycur1 = conn.cursor()  
   mycur2 = conn.cursor()  
   mycurEmployee = conn.cursor()  
   empStr = "E00"  
   queryEmployeeId = "SELECT COUNT(EMPLOYEE\_ID) FROM EMPLOYEE"  
   mycurEmployee.execute(queryEmployeeId)  
   eid = mycurEmployee.fetchone()  
   empId = eid[0] + 1  
   employeeId = empStr + str(empId)  
   queryLogin = 'INSERT INTO LOGIN(USERNAME,PASSWORD)VALUES(%s,%s)'  
   valueLogin = (username, password)  
   mycur1.execute(queryLogin, valueLogin)  
   queryEmp = '''INSERT INTO EMPLOYEE(EMPLOYEE\_ID,EMPLOYEE\_Name,USERNAME,EMPLOYEE\_PHONENO,EMPLOYEE\_EMAIL,EMPLOYEE\_Address,EMPLOYEE\_TYPE)VALUES(%s,%s,%s,%s,%s,%s,%s);'''  
   valueEmp = (employeeId, employeeName, username, employeePhone, employeeEmail, employeeAddress, employeeType)  
   mycur2.execute(queryEmp, valueEmp)  
   print()  
   print(' ->Record Inserted !')  
   print()  
   conn.commit()  
   conn.close()  
    
    
  def UpdateQuantity():  
   conn = getDbConnection()  
   mycurUpdQty = conn.cursor()  
   proId = input("Enter the Product Id : ")  
   pQty = int(input(" Enter The Quantity : "))  
   queryUpdQty = "UPDATE INVENTORY SET PRODUCT\_QUANTITY = %s WHERE PRODUCT\_ID = %s"  
   valUpdQty = (pQty, proId,)  
   mycurUpdQty.execute(queryUpdQty, valUpdQty)  
   print()  
   print(" -> Product Quantity has Been Updated !")  
   print()  
   conn.commit()  
   conn.close()  
    
    
  def salesReport():  
   conn = getDbConnection()  
   mycur = conn.cursor()  
   salesReportQuery = 'SELECT\*FROM PURCHASE'  
   mycur.execute(salesReportQuery)  
   data = mycur.fetchall()  
   wfile = open("C:/Users/ukhan/Desktop/SalesReport.txt", 'w')  
   wfile.write(  
   'PRODUCT ID' + ';' + 'CUSTOMER ID' + ';' + 'PAYMENT ID' + ';' + 'SALE DATE' + ';' + 'BILL ID' + ';' + 'BILL AMOUNT' + '\n')  
   for row in data:  
   wfile.write(  
   row[0] + ';' + ' ' + row[1] + ';' + ' ' + row[2] + ';' + ' ' + str(row[3]) + ';' + ' ' +  
   row[5] + ';' + ' ' + str(row[4]) + '\n')  
   conn.commit()  
   conn.close()  
   wfile.close()  
   print(  
   "Report Has Been Generated Successfully ! ")  
   print()  
   askReport = input('Do You Want To Read This File Now ?(Y/N) ')  
   print()  
   if askReport == 'N':  
   return  
   elif askReport == 'Y':  
   rfile = open("C:/Users/ukhan/Desktop/SalesReport.txt", 'r')  
   readdata = rfile.readlines()  
   for i in readdata:  
   data = i.replace(';', ' ')  
   print(data)  
   rfile.close()  
   else:  
   print('Invalid Entry !')  
    
    
  # MAIN EXECUTION LOGING STARTS  
    
  validLogin = 0  
  validLogin = login()  
  CalGoldRate()  
    
  if validLogin == 0:  
    
   print(' Wrong User Name Or Password !')  
    
  elif validLogin == 1:  
   ch = 0  
   print()  
   while True:  
   MangMenu()  
   ch = int(input(' Choose An Option From The Menu -: '))  
   print()  
    
   if ch == 1:  
   addProduct()  
    
   if ch == 2:  
   addSupplier()  
   if ch == 3:  
   listProduct()  
    
   if ch == 4:  
   searchProduct()  
    
   if ch == 5:  
   addCustomer()  
    
   if ch == 6:  
   searchCustomer()  
    
   if ch == 7:  
   purchase()  
    
   if ch == 8:  
   UpdateQuantity()  
    
   if ch == 9:  
   salesReport()  
    
   if ch == 10:  
   addEmployee()  
    
   elif ch == 11:  
   exit()  
    
    
  elif validLogin == 2:  
   ch = 0  
   print()  
   while True:  
   NorMenu()  
   ch = int(input(' Choose An Option From The Menu -: '))  
   print()  
    
   if ch == 1:  
   listProduct()  
    
   if ch == 2:  
   searchProduct()  
    
   if ch == 3:  
   addCustomer()  
    
   if ch == 4:  
   searchCustomer()  
    
   if ch == 5:  
   purchase()  
    
   elif ch == 6:  
   exit()

**VI. Project demonstration**

* Tools/software/ libraries used
* MySql.Connector – To connect MySql to Python
* Reportlab.lib – To generate a Pdf Invoice

