



ACADEMIC YEAR 2022-2023

LEBANESE UNIVERSITY

Faculty of Science

Department of Math & CS

Course: I3306-Database II

Semester:1

# Shop Management System

---

---

---

Course Instructor

Dr. Mohammad Dbouk

Prepared By

Lynn Aouad – 97751

Rana Jaafar – 97456

Lynn El-mesmar - 97068

# Acknowledgments

---

We would like to express our deepest appreciation to all those who provided us the possibility to complete this project, people who gave their unending support right from the stage the project idea was conceived.

*"To be successful, the first thing to do is to fall in love with your work"*

The project was done with full passion in order to be able to grab the needed attention.

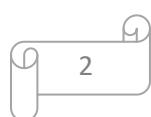
A special gratitude we give to **Dr. Mohammad Dbouk**, for the guidance, inspiration and contribution in stimulating suggestions and encouragement, helped us to coordinate our project especially in writing this report, we are very thankful.

Thank you for your dedication and hard work.

❖ **Rana Jaafar**

❖ **Lynn Aouad**

❖ **Lynn El-mesmar**



# Introduction

---

The internet has revolutionized the way we shop. Because of the numerous advantages and benefits, more and more people these days prefer buying things online over the conventional method of going into stores.

Customers can purchase items from the comfort of their own homes or workplace. Shopping is made easier and convenient for the customer through the internet. It is also easy to cancel the transactions.

## Why shop Online:

- Saves time and efforts.
- The convenience of shopping at home.
- Wide variety/range of products are available.
- Good discounts / lower prices.
- Get detailed information about the product.
- We can compare various models/brands.
- Generally, in physical stores, the sales representatives try to influence the buyers to buy the product. While in online shopping, you're free to do as you will.
- Online consumers can track the order status and delivery status tracking of shipping is also available.
- Customers do not have to stand in queues in cash counters to pay for the products that have been purchased by them. They can shop from their home or workplace and do not have to spend time travelling.

# Chapter I: Application Architecture

In this project we have developed two Web applications:

- User Interface
- Administrator Interface

## We have developed this project using the below technology

- HTML: Page layout has been designed in HTML.
- CSS: used for all the designing part with other frameworks.
- JavaScript: All validation task and animations has been developed by JavaScript.
- PHP: All the business logic has been implemented with PHP.
- Microsoft SQL Server: has been used as database for the project.
- XAMP Server: For executing the PHP scripts and opening the websites.

## User Interface:

We have developed a user interface for the shop. The user will need first to access the website, all the products will be displayed in categories, then he will add to cart all the products that he wants to buy, he can remove from cart also. When the user wants to complete the purchase, he will check out, all the calculations are made including if there is a discount or not, then the total amount is displayed on the screen, if the user wants to continue he will need to insert all his billing details, then place the order.

The order and costumer details will be updated into the database and the administrator system.

## Admin Interface:

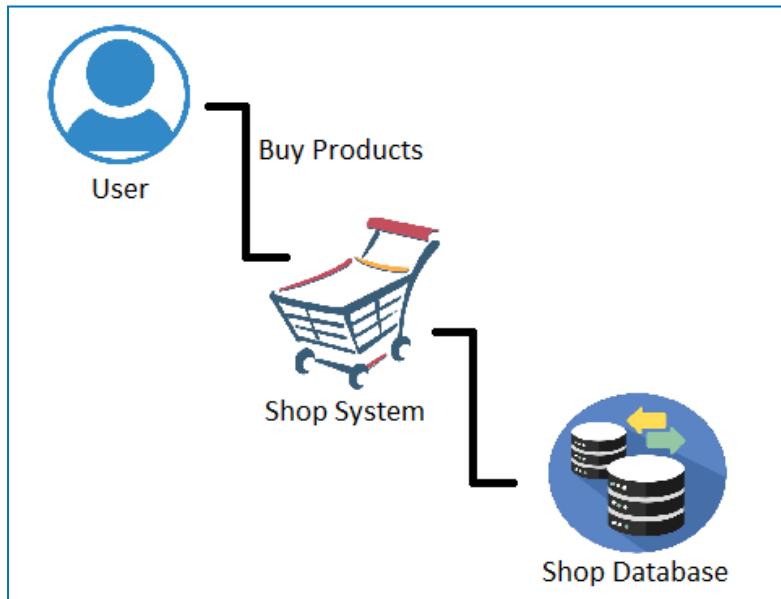
We have developed a Shop Management System. The basic concept to develop this system was to manage shop products, orders, sales and manage all the functionalities related to the shop...

The admin first will need to login then he will be able to:

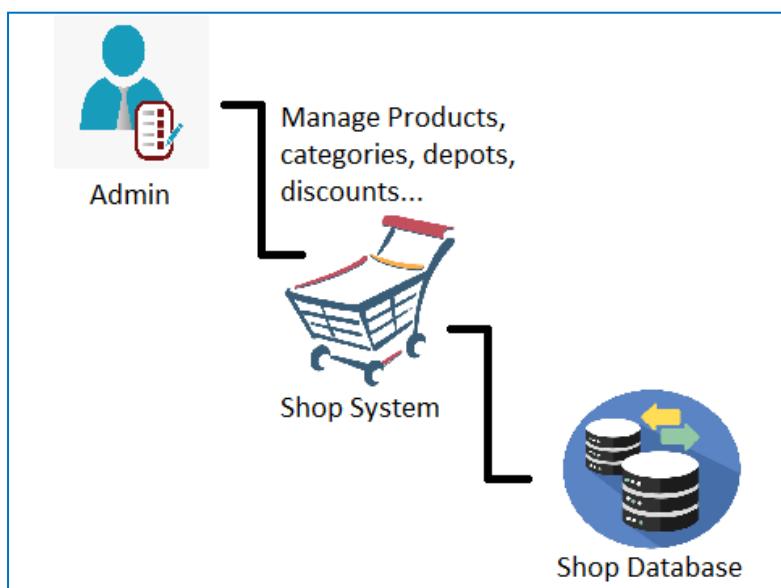
- Manage products sales
- Manage products stocks
- Manage products categories
- Manage depots
- Manage products discounts
- Manage suppliers and their supplies
- View all orders information
- View customers

All those changes will be updated directly into the database.

## Architecture Part 1



## Architecture Part 2



# Chapter II: Database Modeling

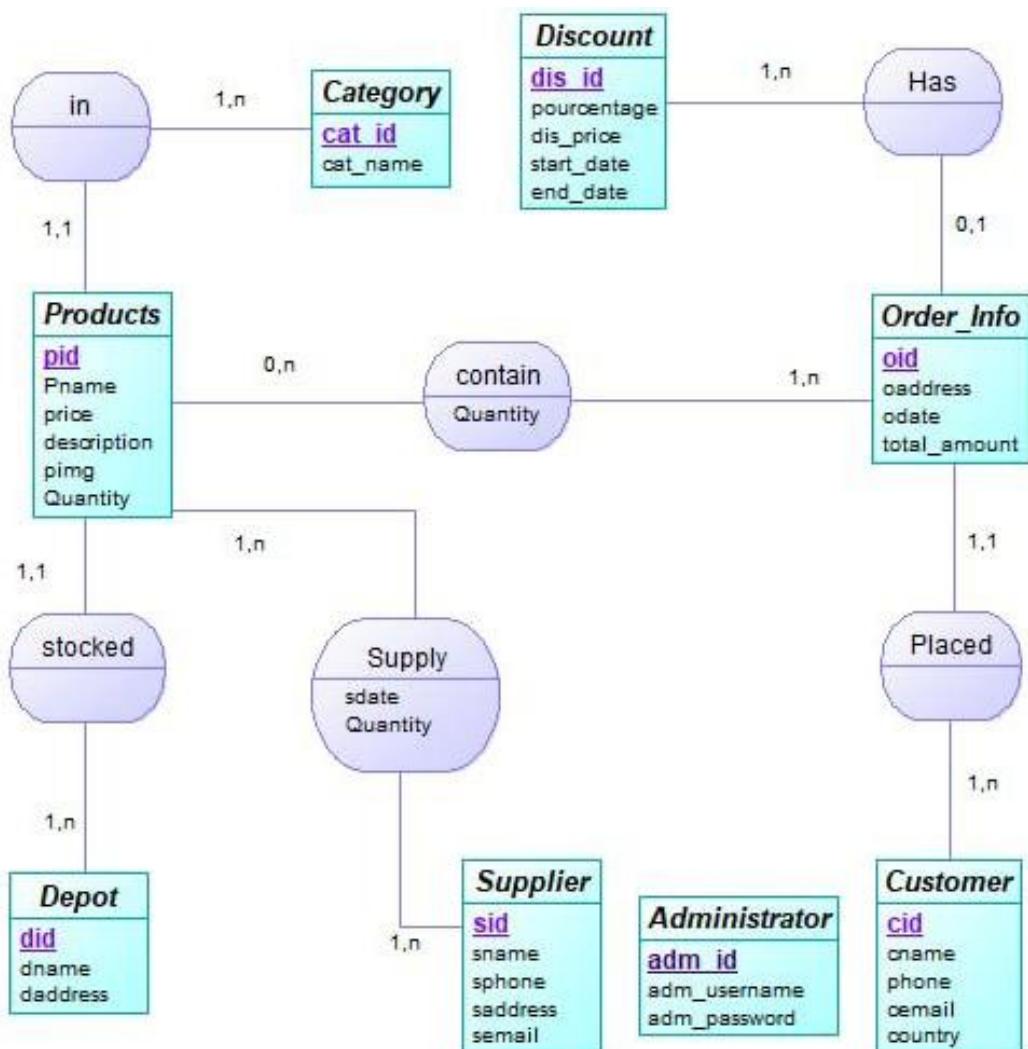
For our project we used power Designer to design the ER and physical diagram that helped us with the creation of database and the generation of the tables and indexes.

## ER Diagram:

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system.

In the ER diagram we have 8 entities, each entity has its own relations:

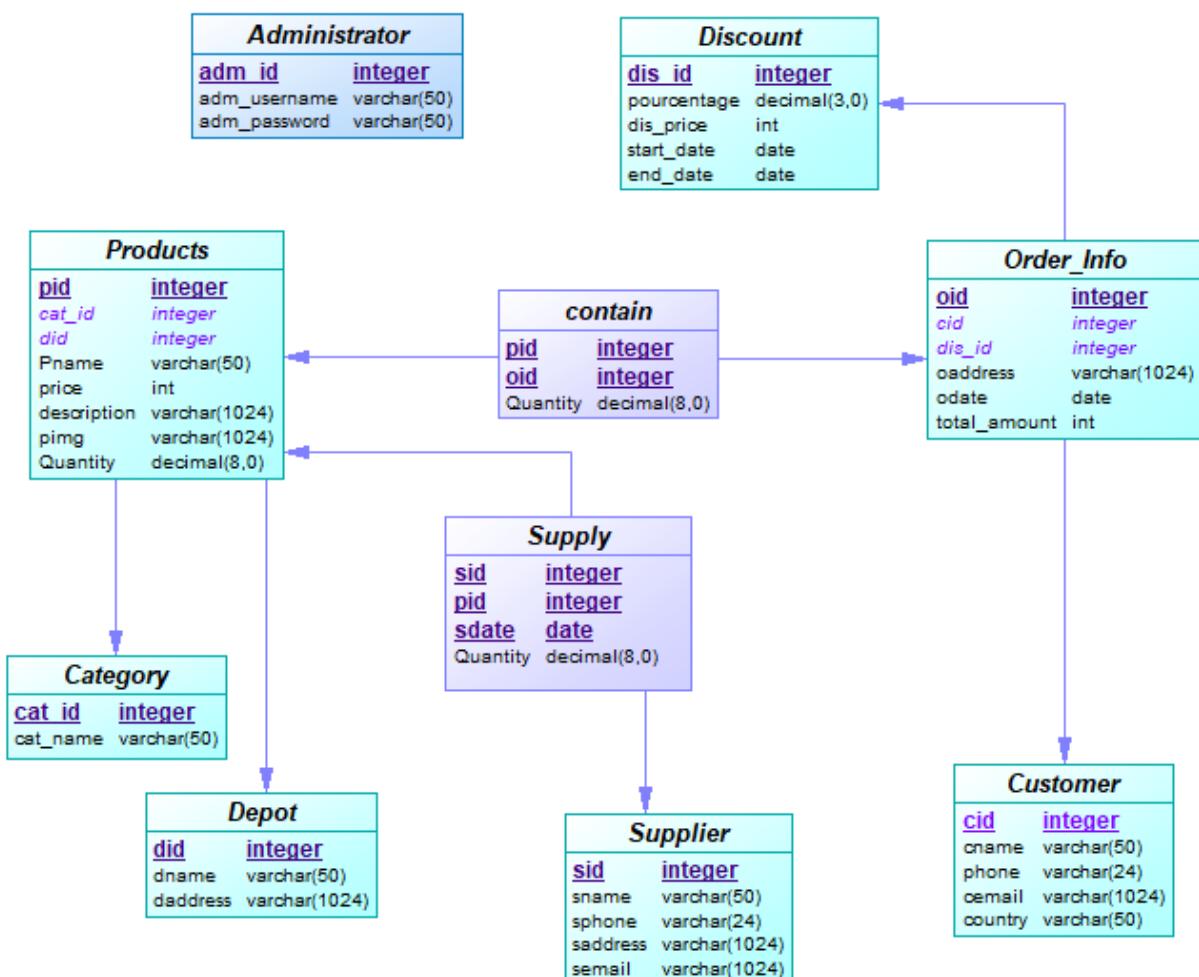
- **Products:** It contains the pid as primary key and has one to one relation with Category entity and Depot entity, and has one to many relation with Supplier entity, and has zero to many relation with Order\_Info entity. Furthermore, a product can be in only one specific Category and stocked in one specific Depot, in addition, a product can be supplied by at least one or many suppliers, and can be contained in zero or many orders .
- **Category:** It contains the cat\_id as primary key and has one to many relation with Products entity. Furthermore, a category can contain at least one or many products.
- **Depot:** It contains the did as primary key and has one to many relation with Products entity. Furthermore, a depot can stock at least one or many products.
- **Supplier:** It contains the sid as primary key and has one to many relation with Products entity. Furthermore, a supplier can supply at least one or many products.
- **Order\_Info:** It contains the oid as primary key and has one to one relation with Customer entity and has one to many relation with Products entity, and has zero to one relation with Discount entity. Furthermore, an order can be ordered only by one specific Customer, in addition, an order can contain at least one or many products, and can have a discount or not .
- **Discount:** It contains the dis\_id as primary key and has one to many relation with Order\_Info entity. Furthermore, a discount can be on at least one or many orders.
- **Customer:** It contains the cid as primary key and has one to many relation with Order\_Info entity. Furthermore, a customer can order at least one or many orders.
- **Administrator:** It contains the adm\_id as primary key, and all the login information about an admin.



## PDM (Physical-Data-Model):

The previous ER diagram was generated to produce a physical diagram that is shown below:

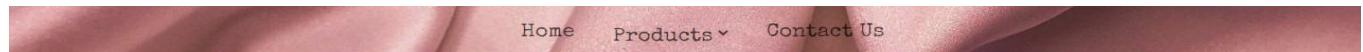
- Category (cat\_id, cat\_name)
- Depot (did, dname, daddress)
- Products (pid, #cat\_id, #did, pname, price, description, pimg, quantity)
- Customer (cid, cname, phone, cemail, country)
- Supplier (sid, sname, sphone, saddress, semail)
- Supply (#sid, #pid, sdate, quantity)
- Discount (dis\_id, pourcentage, dis\_price, start\_date, end\_date)
- Order\_Info (oid, #cid, #dis\_id, oaddress, odate, total\_amount)
- Contain (#pid, #oid, quantity)
- Administrator(adm\_id, adm\_username, adm\_password)



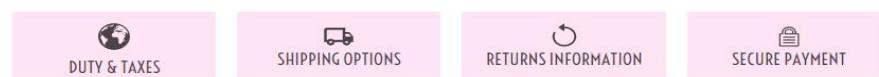
# Chapter III: Services and Interface Modeling

## User Interface:

### 1) Main Page



VICTORIA'S SECRET



### LATEST PRODUCTS



## ABOUT US



### Victoria's Secret does not test on animals

Victoria's Secret & Co. is against animal testing, and no branded products, formulations or ingredients are tested on animals. To be clear, as of April 2021, all personal care products that we sell in China are made in China to avoid animal testing. Victoria's Secret & Co. personal care products sold in the rest of the world are produced in North America and Europe.

UP TO 30% OFF

Have you heard? The event of the season is in full swing with markdowns on your favorite perfumes and body splashes.



### ✉ EMAIL SUBSCRIPTION

Get the inside scoop from Victoria's Secret and Victoria's Secret PINK on exclusive online and in-store offers, new product alerts, store events and store openings in your area. You can unsubscribe any time. View [Contact Info](#) and [Privacy Policy](#)



## Cart is Empty



← Cart (0)



No products in the cart.

### 2) User choose a product category



Products ▾

Winter

Summer

Sets

VICTORIA'S SECRET



DUTY & TAXES



SHIPPING OPTIONS



RETURNS INFORMATION



SECURE PAYMENT

## SUMMER COLLECTION

### Pure Seduction



Price: 40 \$

Description

Winter Sky by Victoria's Secret is an Aromatic fragrance for women. Winter Sky was launched in 2020.

Select Quantity

- | +

ADD TO CART

### Velvet Petals



Price: 40 \$

Description

Platinum Ice by Victoria's Secret is a Fruity Floral fragrance for women. Platinum Ice was launched in 2020.

Select Quantity

- | +

ADD TO CART

### Spring Poppies



Price: 40 \$

Description

Wonderland Woods by Victoria's Secret is a Floral Green fragrance for women. Wonderland Woods Sky was launched in 2020.

Select Quantity

- | +

ADD TO CART

### Blushing Berry Magnolia



Price: 40 \$

Description

Fresh Snowfall by Victoria's Secret is a Floral fragrance for women. Fresh Snowfall was launched in 2020.

Select Quantity

- | +

ADD TO CART

## 3) Add products into Cart

### SUMMER COLLECTION

#### Pure Seduction



Price: 40 \$

Description

Winter Sky by Victoria's Secret is an Aromatic fragrance for women. Winter Sky was launched in 2020.

Select Quantity

- | 3 | +

ADD TO CART

#### Velvet Petals



Price: 40 \$

Description

Platinum Ice by Victoria's Secret is a Fruity Floral fragrance for women. Platinum Ice was launched in 2020.

Select Quantity

- | +

ADD TO CART

SUMMER COLLECTION

**Pure Seduction**

Price: 40 \$

Description

Winter Sky by Victoria's Secret is an Aromatic fragrance for women. Winter Sky was launched in 2020.

Select Quantity

- | 3 | +

ADD TO CART

**Pure Seduction**

Price: 40 \$

Description

Winter Sky by Victoria's Secret is an Aromatic fragrance for women. Winter Sky was launched in 2020.

3 Pure Seduction were added to cart successfully

OK

#### 4) View Cart – User can remove product from cart

**Cart (4)**

Pure Seduction	Qty 3	40 \$
Spring Poppies	Qty 1	40 \$
Dreamy Plum Dahlia	Qty 1	40 \$
Red Temptation	Qty 1	120 \$

Cart Total: 320 \$

Discount: 20% \$

Shipping: 5 \$

Total: 261 \$

Payment: Cash on delivery

**Checkout**

#### 5) Checkout & Place Order

##### Billing Details

Name\*

Country\*

Address\*

Email address

We'll inform you about the tracking informations via Email or mobile number.

Mobile Number\*

Your personal data will be used to process your order and support your experience throughout this website.

I have read and agree to the website **terms and conditions** \*

**Place Order**

##### Billing Details

Name\*

Country\*

Address\*

Email address

Mobile Number\*

Your personal data will be used to process your order and support your experience throughout this website.

I have read and agree to the website **terms and conditions** \*

**Place Order**

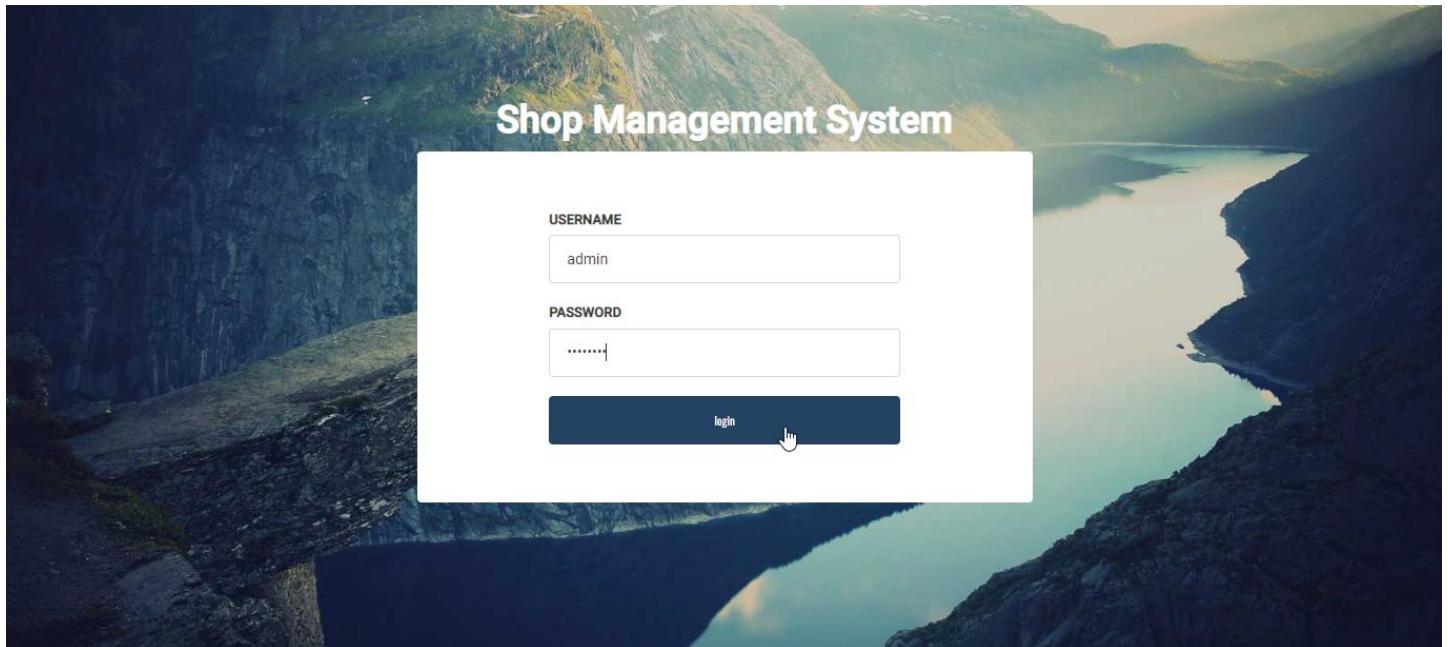
**Thank you for your order!**

Your order has been received, and it's now being processed.

**OK**

## Administrator Interface:

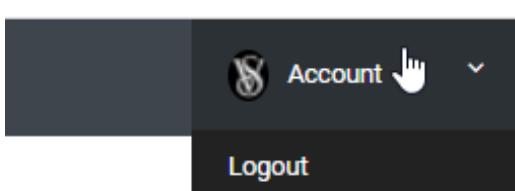
### 1) Login



### 2) Dashboard – All sections admin can add, remove update...

The dashboard page has a dark header with the 'Shop Management System' logo and an 'Account' dropdown. On the left is a sidebar with links: 'Dashboard' (highlighted with an arrow), 'Products', 'Out Of Stock', 'Categories', 'Depots', 'Suppliers', 'Supplies', and 'Discounts'. The main area is titled 'Dashboard' and features three summary boxes: 'TOTAL CUSTOMERS' (13), 'TOTAL ORDERS' (14), and 'TOTAL MONEY' (4934\$). Arrows point from the text to their respective boxes. Below these are sections for 'CUSTOMERS COUNTRIES' and 'Customers Statistics'. The 'CUSTOMERS COUNTRIES' section shows a table with data for Lebanon, France, USA, Italy, and Germany. The 'Customers Statistics' section shows a message: 'Showing 1 to 1 of 1 entries' with arrows pointing to the numbers 1 and 1.

Logout



### 3) Products Section

Shop Management System

MAIN

- Dashboard
- + Products**
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

8	Velvet Petals			200	40\$	fragrance for women. Winter Sky was launched in 2020.					
9	Spring Poppies			200	40\$	Platinum Ice by Victoria's Secret is a Fruity Floral fragrance for women. Platinum Ice was launched in 2020.	74	2000			
10	Blushing Berry Magnolia			200	40\$	Wonderland Woods by Victoria's Secret is a Floral Green fragrance for women. Wonderland Woods Sky was launched in 2020.	27	2000			
Showing 1 to 10 of 18 entries											
<input type="button" value="PREVIOUS"/> <input type="button" value="1"/> <input type="button" value="2"/> <input type="button" value="NEXT"/>											

Get number of products in a specific depot

Depot id  go

remove

Update Product

Shop Management System

MAIN

- Dashboard
- + Products**
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

EDIT PRODUCT

Product Name	Blushing Berry Magnolia
Product Image	<input type="text" value="./img/v4.jpeg"/>
Category ID	200
Price	40
Description	Fresh Snowfall by Victoria's Secret is a Floral for women. Fresh Snowfall was launched in 2020.
Quantity	76
Depot ID	2000

#### 4) Out-Of-Stock

Shop Management System

MAIN

- Dashboard
- + Products
  - Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

Out Of Stock - Total items: 2

OUT OF STOCK PRODUCTS DETAILS

Show 10 entries	Search:		
pid	pimg	pname	price(\$)
5		Love Spell Frosted	20
18		Aqua Kiss	120

Showing 1 to 2 of 2 entries

PREVIOUS 1 NEXT

#### 5) Category Section

Shop Management System

MAIN

- Dashboard
- + Products
  - Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

Category

ADD CATEGORIES

Category Name	category4
---------------	-----------

Add category

ALL CATEGORIES DETAILS

Show 10 entries	Search:	
cat_id	cat_name	Action
100	winter	
200	summer	
300	sets	

Showing 1 to 3 of 3 entries

PREVIOUS 1 NEXT

Shop Management System

MAIN

- Dashboard
- + Products
  - Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

Edit Category

EDIT CATEGORIES

Category ID	200
Course Name	summer

Update Cancel

## 6) Depots Section

Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots**
- Suppliers
- Supplies
- Discounts

Depot

ADD DEPOT

Depot Name: depot4

Depot Address: address\_3

Add Depot

ALL DEPOTS DETAILS

did	dname	address	Action
1000	depot1	address_1	<input checked="" type="checkbox"/> <input type="button" value="x"/>
2000	depot2	address_2	<input checked="" type="checkbox"/> <input type="button" value="x"/>
3000	depot3	address_1	<input checked="" type="checkbox"/> <input type="button" value="x"/>

Showing 1 to 3 of 3 entries

PREVIOUS 1 NEXT

Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

Edit Depot

EDIT DEPOTS

Depot ID: 3000

Depot Name: depot3

Depot Address: address\_1

Update Cancel

ALL DEPOTS DETAILS

did	dname	address	Action
1000	depot1	address_1	<input checked="" type="checkbox"/> <input type="button" value="x"/>
2000	depot2	address_2	<input checked="" type="checkbox"/> <input type="button" value="x"/>
3000	depot3	address_1	<input checked="" type="checkbox"/> <input type="button" value="x"/>

Showing 1 to 3 of 3 entries

PREVIOUS 1 NEXT

## 7) Suppliers Section

Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots
- Suppliers**
- Supplies
- Discounts

Supplier

ALL SUPPLIERS DETAILS

sid	sname	stel	semail	saddress	Action
1	Fadi Maalouf	3825853	fadimaalouf@gmail.com	Beirut	<input checked="" type="checkbox"/> <input type="button" value="x"/>
2	Tony Nakour	71523980	tonynakour@gmail.com	Saida	<input checked="" type="checkbox"/> <input type="button" value="x"/>
3	Alaa Ghodban	81200756	alaaghodban@gmail.com	Tripoli	<input checked="" type="checkbox"/> <input type="button" value="x"/>

Showing 1 to 3 of 3 entries

PREVIOUS 1 NEXT

ADD SUPPLIERS

Supplier Name:

MAIN

 Dashboard Products Out Of Stock Categories Depots Suppliers Supplies Discounts

1	Fadi Maalouf	3825853	fadimaloul@gmail.com	Beirut	 
2	Tony Nakour	71523980	tonynakour@gmail.com	Saida	 
3	Alaa Ghodban	81200756	alaaghodban@gmail.com	Tripoli	 

Showing 1 to 3 of 3 entries

PREVIOUS 1 NEXT

## ADD SUPPLIERS

Supplier Name	<input type="text" value="supplier"/>
Phone Number	<input type="text" value="03698457"/>
Email Address	<input type="text" value="supplier@gmail.com"/>
Address	<input type="text" value="supplier_address"/>

MAIN

 Dashboard Products Out Of Stock Categories Depots Suppliers Supplies Discounts

## Edit Supplier

## EDIT SUPPLIERS

Supplier ID	<input type="text" value="3"/>
Supplier Name	<input type="text" value="Alaa Ghodban"/>
Phone Number	<input type="text" value="81200756"/>
Email Address	<input type="text" value="alaaghodban@gmail.com"/>
Address	<input type="text" value="Tripoli"/>

## 8) Supplies Section

Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies**
- Discounts

Fadi Maalouf	1	Platinum ice	2	2021-2-1	40	
Fadi Maalouf	1	Cheers Again	6	2021-5-7	215	
Fadi Maalouf	1	Pure Seduction	7	2021-2-28	24	
Fadi Maalouf	1	Velvet Petals	8	2021-3-2	76	
Fadi Maalouf	1	Velvet Petals Decadent	11	2021-10-1	10	
Fadi Maalouf	1	Dreamy Plum Dahlia	12	2022-4-15	13	
Fadi Maalouf	1	Bare Vanilla	13	2021-6-19	50	
Fadi Maalouf	1	Coconut Passion	14	2022-1-15	45	
Fadi Maalouf	1	Red Temptation	15	2022-4-15	14	

Showing 1 to 10 of 17 entries

PREVIOUS 1 2 NEXT

Get total number of today's supplies for a specific supplier

Supplier id  go

Transfer Supplies

From Supplier id  To Supplier id  Transfer

Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

Supplier id  go

Transfer Supplies

From Supplier id  To Supplier id  Transfer

ADD SUPPLIES

Supplier ID: 1  
Product ID: 3  
Date: 2022-1-15  
Quantity: 100

Add Supply

Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

Edit Supply

EDIT SUPPLIES

Supplier ID: 1  
Product ID: 1  
Date: 2021-1-2  
Quantity: 147

Update Cancel

## 9) Discounts Section

Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts**

Discount

**ADD DISCOUNT**

%	40
price	600
startDate	04/16/2022
endDate	04/27/2022

**Add discount**

**ALL DISCOUNTS DETAILS**

dis_id	%	dis_price	start_date	end_date
1	10%	100\$	2022-1-11	2022-5-11
2	20%	300\$	2022-1-11	2022-5-11

Showing 1 to 2 of 2 entries

PREVIOUS 1 NEXT

Delete all out dated Discounts! ↩

**Delete**

## 10) Customers Details

Dashboard



Shop Management System

MAIN

- Dashboard
- Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

Customers

**ALL CUSTOMERS**

cid	cname	phone	cemail	country
1	Lynn Aouad	71658432	lynnaouad@gmail.com	Lebanon
2	Rana Jaafar	03697541	ranajaafar@gmail.com	Lebanon
3	Lynn Elmesmar	71659948	lynnelmesmar@gmail.com	Lebanon
4	Maya Assi	03695547	mayaassi@gmail.com	France
5	Joelle Haddad	81365447	joellehaddad@gmail.com	USA
6	Norma kawas	03669874	normakawas@gmail.com	USA
7	Lea hassoun	81365447	leahassoun@gmail.com	USA
8	Maya hassoun	03669874	mayahassoun@gmail.com	USA
9	Marita hellani	03669874	maritahellani@gmail.com	Italy
10	Savine flores	03669874	savineflores@gmail.com	Germany
11	customer test	03789456	customer@gmail.com	ITALY
12	rfszrgf srgsrgreg	13343223454	rana@gmail.com	Lebanon

19

## 11) Order Info

### Dashboard



Shop Management System

Account

MAIN

- Dashboard
- + Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

	4839	5	1	Joelle Haddad	2022-1-28	USA	195\$	eye
	4840	6	1	Norma kawas	2022-2-2	USA	271\$	eye
	4841	9	1	Marita hellani	2022-2-15	Italy	461\$	eye
	4842	10	2	Savine flores	2022-2-16	Germany	653\$	eye
	4843	1	0	Lynn Aoaud	2022-1-27	Lebanon	85\$	eye
	4844	1	2	Lynn Aoaud	2022-2-4	Lebanon	347\$	eye
	4845	1	2	Lynn Aoaud	2022-2-26	Lebanon	653\$	eye
	4846	11	2	customer test	2022-4-14	italy	437\$	eye
	4847	12	0	rfszrgf srgsrgreg	2022-4-15	ddd	25\$	eye
	4848	13	2	SFS aouad	2022-4-16	beirut	261\$	eye

Get number of orders of a specific customer →  go

Get number of Today's Orders →  go

Get number of orders of a specific date →  go

View order details

Shop Management System

Account

MAIN

- Dashboard
- + Products
- Out Of Stock
- Categories
- Depots
- Suppliers
- Supplies
- Discounts

### Order Details

**Order id:** 4838

**Order date:** 2022-1-28

**Order address:** France

**Order:**

Wonderland Woods(x2) : 40\$  
 Spring Poppies(x3) : 120\$  
 Coconut Passion(x1) : 120\$

**Discount:** 10%

**Delivery:** 5\$

**Total amount:** 537\$

**Customer:** Maya Assi

**Customer phone:** 03695547

**Customer email:** mayaassi@gmail.com

**Customer country:** France

# Chapter IV: Implementation and Testing

## Database Connection:

```
<?php
    $servername="HP-PC\SQLEXPRESS";
    $connectionInfo = array("Database"=>"shop", "UID"=>"USER", "PWD"=>"SHOP");

    $conn = sqlsrv_connect($servername,$connectionInfo);

    if(!$conn)
        die("Connection failed" .sqlsrv_errors());
?>
```

User connection

```
<?php
$servername = "HP-PC\SQLEXPRESS";
$connectionInfo = array("Database" => "shop", "UID" => "ADMIN", "PWD" => "admin123");

$conn = sqlsrv_connect($servername, $connectionInfo);

if (!$conn)
    die("Connection failed" . sqlsrv_errors());
?>
```

Admin connection

## Some commands used in coding:

When user place order we call the **PLACE\_ORDER procedure**, then we insert each product in Contain table.

```
//CALLING A PROCEDURE TO PLACE THE ORDER (add the client info and the order info)
$dis_id=$_SESSION['discount'];
$price_final = $_SESSION['price_final'];

$parameters = array( array($name,SQLSRV_PARAM_IN) ,
                    array($phone,SQLSRV_PARAM_IN),
                    array($email,SQLSRV_PARAM_IN),
                    array($country,SQLSRV_PARAM_IN),
                    array($dis_id,SQLSRV_PARAM_IN),
                    array($address,SQLSRV_PARAM_IN));

$call_proc = "{CALL PLACE_ORDER(?,?,?,?,?,?)}";
$stmtt = sqlsrv_query($conn, $call_proc, $parameters);

if($stmtt === false) {
    if( ($errors = sqlsrv_errors()) != null){
        foreach($errors as $error{
            echo "sqlstate:".$error['SQLSTATE']."<br>";
            echo "code:".$error['code']."<br>";
            echo "message:".$error['message']."<br>";
            echo "stmt error";}}}
```

```
for($k=0; $k<$lenght; $k++){
    $query= "insert into Contain VALUES(\".$data[$k]['id'].\", \"$oid.\", \"$data[$k]['qty'].'\");

    $stmtt2 = sqlsrv_query($conn, $query);

    if($stmtt2 === false) {
        if( ($errors = sqlsrv_errors()) != null){
            foreach($errors as $error{
                echo "sqlstate:".$error['SQLSTATE']."<br>";
                echo "code:".$error['code']."<br>";
                echo "message:".$error['message']."<br>";
                echo "stmt error";}}}
```

```

|CREATE PROCEDURE PLACE_ORDER
|@CNAME      varchar(50),
|@PHONE       varchar(24),
|@CEMAIL      VARCHAR(1024),
|@COUNTRY     varchar(50),
|@DIS_ID      INT,
|@OADDRESS    varchar(1024)
|AS BEGIN
|    declare @cid int
|    begin transaction
|        exec dbo.ADD_CUSTOMER @CNAME,@PHONE,@CEMAIL,@COUNTRY,@cid out
|        exec dbo.ADD_ORDER @CID,@DIS_ID, @OADDRESS
|        commit
|    END
|    go

```



```

|create procedure ADD_CUSTOMER
|@CNAME      varchar(50),
|@PHONE       varchar(24),
|@CEMAIL      varchar(1024),
|@COUNTRY     varchar(50),
|@CID         int   out
|as BEGIN
|
|    if exists(select * from Customer where cname=@cname and PHONE=@PHONE and cemail=@cemail and country=@country)
|        set @CID=(select cid from Customer where cname=@cname and PHONE=@PHONE and cemail=@cemail and country=@country)
|    else
|        BEGIN
|            Insert into Customer(CNAME,PHONE,CEMAIL,COUNTRY)
|                    values(@CNAME,@PHONE,@CEMAIL,@COUNTRY);
|
|            set @CID=(select max(CID) from Customer);
|        end
|    end
|go

-----
create procedure ADD_ORDER
|@CID          int ,
|@DIS_ID       int,
|@OADDRESS     varchar(1024)

as BEGIN
    declare @YY VARCHAR(5),@MM VARCHAR(5),@DD VARCHAR(5)
    declare @date_0 varchar(30), @OID int

    SET @YY=(select YEAR(GETDATE()))
    SET @MM=(select MONTH(GETDATE()))
    SET @DD=(select DAY(GETDATE()))
    set @date_0= @YY+'-'+@MM+'-'+@DD

    insert into Order_Info(CID,OADDRESS,DIS_ID,ODATE,TOTAL_AMOUNT)
                    values(@CID,@OADDRESS,@DIS_ID,@DATE_0,0);
END
go

```

When user add products into Cart we call the **FETCH\_DIS procedure**, to know if there is a discount.

```
$parameters = array( array($carttotal,SQLSRV_PARAM_IN) ,
    array(&$price_final,SQLSRV_PARAM_OUT),
    array(&$pourcentage,SQLSRV_PARAM_OUT),
    array(&$dis_id,SQLSRV_PARAM_OUT));
$call_proc = "{CALL FETCH_DIS(?, ?, ?, ?)}";
$stmtt = sqlsrv_query($conn, $call_proc, $parameters);

// to put the result in the out parameter
if(sqlsrv_next_result($stmtt)==false){
    if( ($errors = sqlsrv_errors()) != null){
        foreach($errors as $error){
            echo "sqlstate:".$error['SQLSTATE']."<br>";
            echo "code:".$error['code']."<br>";
            echo "message:".$error['message']."<br>";
            echo "result error";}}}

$_SESSION["discount"]=$dis_id;
$_SESSION["price_final"]=$price_final;
```

```
create procedure FETCH_DIS
@price int,
@price_final int out,
@pourcentage decimal(3,0) OUT,
@ID int OUT
AS
BEGIN
declare crs_1 cursor for select dis_id,pourcentage,dis_price from dbo.DIS
declare @dis_id int ,@pr decimal(3,0) ,@dis_price int ,@dis_id1 int ,@pr1 int ,@delivery int
set @delivery = 5

open crs_1
fetch next from crs_1 into @dis_id, @pr , @dis_price

while @@fetch_status=0
BEGIN
    if(@dis_id1>=0)
        BEGIN
            set @pr1=(select dis_price from Discount where dis_id=@dis_id1) -- dis_price for the discount before!
            if(@price >= @dis_price)
                BEGIN
                    if(@pr1<@dis_price)
                        set @dis_id1=@dis_id
                end
            end
        else
            BEGIN
                if(@price >= @dis_price)
                    set @dis_id1 = @dis_id
            end
        end
    fetch next from crs_1 into @dis_id,@pr,@dis_price
end

if(@dis_id1>=0)
    BEGIN
        set @pr= (select pourcentage from discount where dis_id=@dis_id1)
        set @price_final= (@price-(@pr/100)*@price)+@delivery
        set @ID=@DIS_ID1
        set @pourcentage = @pr
    end

CLOSE crs_1
deallocate crs_1
end
go
```

When admin transform all supplies from supplier1 to supplier 2, we call the **TRANSFORME** procedure.

```
<?php

if(isset($_POST['transfer'])){
    $s1 = trim($_POST['s1']);
    $s2 = trim($_POST['s2']);

    $sql = "SELECT count(*) as count from supplier s1, supplier s2 where s1.sid='".$s1." AND s2.sid='".$s2."'";
    $result = sqlsrv_query($conn, $sql);
    $row=sqlsrv_fetch_array($result, SQLSRV_FETCH_ASSOC);

    if($row['count'] == 1){

        $parameters = array( array($s1,SQLSRV_PARAM_IN) ,
        array($s2,SQLSRV_PARAM_IN));
```



```
        $call_proc = "{CALL TRANSFORME(?,?)}";
        $stmtt = sqlsrv_query($conn, $call_proc, $parameters);

        if(sqlsrv_next_result($stmtt)==false){
            if( ($errors = sqlsrv_errors()) != null){
                foreach($errors as $error){
                    echo "sqlstate:".$error['SQLSTATE']."<br>";
                    echo "code:".$error['code']."<br>";
                    echo "message:".$error['message']."<br>";
                    echo "result error";}}}
        }
        else{
            echo "<script> alert('Please enter a valid sid!')</script>";
        }
    }
    ?>
```

```
create procedure TRANSFORME
    @sid int,
    @sid1 int
as
BEGIN
    if(@sid!=@sid1)
    begin
        if ((EXISTS(select * from supplier where sid=@sid1)) and (exists(select * from supplier where sid=@sid)))
        begin
            begin transaction
            declare cursortr CURSOR
            for select * from supply where sid=@sid
            declare @pid int,@date_s date,@qt decimal(8,0)
            open cursortr
            fetch next from cursortr into @sid,@pid,@date_s,@qt
            while @@fetch_status=0
            BEGIN
                if exists(select * from supply where pid=@pid and sid=@sid1 and sdate=@date_s)
                begin
                    update supply set QUANTITY=QUANTITY+@QT    where pid=@pid and sid=@sid1 and sdate=@date_s
                    delete supply where pid=@pid and sid=@sid and sdate=@date_s and QUANTITY=@QT
                end
                else
                BEGIN
                    insert into supply VALUES(@sid1,@pid,@date_s,@qt)
                    delete supply where pid=@pid and sid=@sid and sdate=@date_s AND QUANTITY=@QT
                end
                fetch next from cursortr into @sid,@pid,@date_s,@qt
            end
            CLOSE cursortr
            deallocate cursortr
            commit
        end
    end
end
go
```

In Dashboard we used multiple functions:

CUSTOMERS COUNTRIES						
Show 10 entries		Search:				
Lebanon	France	USA	Italy	Germany		
38%	7%	30%	15%	7%		

Showing 1 to 1 of 1 entries

PREVIOUS 1 NEXT



```
<?php
$sql1 = "SELECT [dbo].[country_pourcentage]('Lebanon') as pr1";
$result1 = $sqlsrv_query($conn, $sql1);
$row1 = $sqlsrv_fetch_array($result1, SQLSRV_FETCH_ASSOC);

$sql2 = "SELECT [dbo].[country_pourcentage]('France') as pr2";
$result2 = $sqlsrv_query($conn, $sql2);
$row2 = $sqlsrv_fetch_array($result2, SQLSRV_FETCH_ASSOC);

$sql3 = "SELECT [dbo].[country_pourcentage]('USA') as pr3";
$result3 = $sqlsrv_query($conn, $sql3);
$row3 = $sqlsrv_fetch_array($result3, SQLSRV_FETCH_ASSOC);

$sql4 = "SELECT [dbo].[country_pourcentage]('Italy') as pr4";
$result4 = $sqlsrv_query($conn, $sql4);
$row4 = $sqlsrv_fetch_array($result4, SQLSRV_FETCH_ASSOC);

$sql5 = "SELECT [dbo].[country_pourcentage]('Germany') as pr5";
$result5 = $sqlsrv_query($conn, $sql5);
$row5 = $sqlsrv_fetch_array($result5, SQLSRV_FETCH_ASSOC);
```

```
-- get customers country percentage
create function country_pourcentage(@country varchar(50))
returns INT
BEGIN
declare @count integer,@pr integer,@RES decimal(5,2)
set @count=(select count(*)
            from CUSTOMER)
set @pr=(select count(*)
            from CUSTOMER
            where COUNTRY=@country
            )
set @RES=(@pr*100)/(@count*1.0)
return @RES
end
go
```

## Dashboard



```
$sql = "SELECT [dbo].[total_amount]() as nb";
$result = $sqlsrv_query($conn, $sql);
$row = $sqlsrv_fetch_array($result, SQLSRV_FETCH_ASSOC);
```

```
-- get total amount of all orders
create function total_amount()
returns int
BEGIN
declare @a int
set @a=(select sum(total_amount) from order_info)
return @a
end
go
```

## Creating Tables:

```
create table CATEGORY (
    CAT_ID      int      IDENTITY(100,100)      not null,
    CAT_NAME    varchar(50)           not null,
    constraint PK_CATEGORY primary key nonclustered (CAT_ID)
)

create table CUSTOMER (
    CID        int      IDENTITY(1,1)      not null,
    CNAME     varchar(50)           not null,
    PHONE      varchar(24)          not null,
    CEMAIL    varchar(1024)         not null,
    COUNTRY   varchar(50)           not null,
    constraint PK_CUSTOMER primary key nonclustered (CID)
)

create table DISCOUNT (
    DIS_ID      int      IDENTITY(0,1)      not null,
    POURCENTAGE decimal(3,0)           not null,
    DIS_PRICE   int            not null,
    START_DATE  date           null,
    END_DATE   date           null,
    constraint PK_DISCOUNT primary key nonclustered (DIS_ID)
)

create table ORDER_INFO (
    OID        int      IDENTITY(4835,1)      not null,
    CID        int            not null,
    OADDRESS   varchar(1024)         not null,
    DIS_ID     int            null,
    ODATE      date           not null,
    TOTAL_AMOUNT int           not null,
    constraint PK_ORDER_INFO primary key nonclustered (OID),
    constraint FK_ORDER_IN_PLACED_CUSTOMER foreign key (CID) references CUSTOMER (CID),
    constraint FK_ORDER_IN_HAS_DISCOUNT foreign key (DIS_ID) references DISCOUNT (DIS_ID)
)

create table DEPOT (
    DID        int      IDENTITY(1000,1000)      not null,
    DNAME     varchar(50)           not null,
    DADDRESS   varchar(1024)         not null,
    constraint PK_DEPOT primary key nonclustered (DID)
)

create table PRODUCTS (
    PID        int      IDENTITY(1,1)      not null,
    PNAME     varchar(50)           not null,
    PIMG      varchar(1024)         not null,
    CAT_ID    int            not null,
    PRICE     int            not null
    constraint CKC_PRICE_PRODUCT check (PRICE >= 0),
    DESCRIPTION varchar(1024)         not null,
    QUANTITY   decimal(8,0)          not null
    constraint CKC_QUANTIT_PRODUCT check (QUANTITY >= 0),
    DID        int            not null,
    constraint PK_PRODUCT primary key nonclustered (PID),
    constraint FK_PRODUCT_STOCKED_DEPOT foreign key (DID) references DEPOT (DID),
    constraint FK_PRODUCT_IN_CATEGORY foreign key (CAT_ID) references CATEGORY (CAT_ID)
)

create table CONTAIN (
    PID        int            not null,
    OID        int            not null,
    QUANTITY   decimal(8,0)          not null
    constraint CKC_QUANTITY_CONTAIN check (QUANTITY >= 1),
    constraint PK_CONTAIN primary key (PID, OID),
    constraint FK_CONTAIN_CONTAIN_ORDER_IN foreign key (OID) references ORDER_INFO (OID),
    constraint FK_CONTAIN_CONTAIN2_PRODUCTS foreign key (PID) references PRODUCTS (PID)
)
```

```

|create table SUPPLIER (
    SID          int      IDENTITY(1,1)      not null,
    SNAME        varchar(50)           not null,
    SPHONE       varchar(24)           not null,
    SEMAIL       varchar(1024)          not null,
    SADDRESS      varchar(1024)          null,
    constraint PK_SUPPLIER primary key nonclustered (SID)
)

|create table SUPPLY(
    SID          int      not null,
    PID          int      not null,
    SDATE        date    not null,
    QUANTITY     decimal(8,0)         not null
    constraint CKC_QUANTITY_SUPPLY check (QUANTITY >= 1),
    constraint PK_SUPPLY primary key (SID, PID, SDATE),
    constraint FK_SUPPLY_SUPPLY_PRODUCT foreign key (PID) references PRODUCTS (PID),
    constraint FK_SUPPLY_SUPPLY2_SUPPLIER foreign key (SID) references SUPPLIER (SID)
)

|create table ADMINISTRATOR (
    ADM_ID        int      IDENTITY(1,1)      not null,
    ADM_USERNAME  varchar(50)           not null,
    ADM_PASSWORD  varchar(50)           not null,
    constraint PK_ADMINISTRATOR primary key nonclustered (ADM_ID)
)

```

## Creating Logins:

```

USE [master]
GO
CREATE LOGIN [ADMIN] WITH PASSWORD=N'admin123', DEFAULT_DATABASE=[Shop]
GO
use Shop
GO
|CREATE user [admin] for Login [ADMIN]

grant select on out_of_stock to admin
grant select,update,delete,insert on products to admin
grant select,update,delete,insert on depot to admin
grant select,update,delete,insert on category to admin
grant select,update,delete,insert on discount to admin
grant select,update,delete,insert on supplier to admin
grant select,update,delete,insert on supply to admin
grant select on order_info to admin
grant select on customer to admin
grant select on contain to admin
grant select on administrator to admin
grant exec on DELETE_DIS to admin
grant exec on TRANSFORME to admin

grant exec on country_pourcentage to admin
grant exec on NB_OFSUPPLY to admin
grant exec on NB_OFPRODUCT to admin
grant exec on NB_OFORDER to admin
grant exec on TODAY_ORDERS to admin
grant exec on NB_OFORDER2 to admin
grant exec on total_amount to admin

```

```

USE [master]
GO
CREATE LOGIN [USER] WITH PASSWORD=N'SHOP', DEFAULT_DATABASE=[Shop]
GO
use Shop
GO
|create user [shop] for Login [USER]

grant SELECT ON on_stock TO shop
grant SELECT,UPDATE on order_info to shop
grant SELECT,UPDATE on products to shop
grant SELECT,UPDATE,INSERT on Contain to shop
grant exec on ADD_ORDER to shop
grant exec on ADD_CUSTOMER to shop
grant exec on PLACE_ORDER to shop
grant exec on FETCH_DIS to shop
grant exec on GET_QT to shop

```

## Creating Indexes:

<pre> create index CONTAIN_FK on CONTAIN (     OID ASC ) go</pre>	<pre> create index STOCKED_FK on PRODUCTS (     DID ASC ) go</pre>
<pre> create index PLACED_FK on ORDER_INFO (     CID ASC ) go</pre>	<pre> create index IN_FK on PRODUCTS (     CAT_ID ASC ) go</pre>
<pre> create index HAS_FK on ORDER_INFO (     DIS_ID ASC ) go</pre>	<pre> create index SUPPLY2_FK on SUPPLY (     PID ASC ) go</pre>

## Creating Functions:

```
-- get quantity in stock of a specific product
|create function GET_QT(@PID int)
returns int
BEGIN
    declare @Q int
    if((select count(*) from products where pid=@pid)=1)
    BEGIN
        set @Q=(select quantity FROM products WHERE pid=@PID)
    END

    return @Q
end
go

-- get number of supplies of a specific supplier
|create function NB_OFSUPPLY(@sid int)
returns int
BEGIN
    declare @nb int
    set @nb=(select count(*) from supply
        where sid=@sid AND sdate=(cast((select YEAR(GETDATE())) as varchar(50))+ '-' +
            cast(( select MONTH(GETDATE())) as varchar(50))+'-' +
            cast((select DAY(GETDATE())) as varchar(50))))
    return @nb
end
go

-- get number of today's orders
|create function TODAY_ORDERS()
returns int
BEGIN
    declare @nb int
    set @nb=(select count(*) from ORDER_INFO
        where odate=(cast((select YEAR(GETDATE())) as varchar(50))+
            +'-' +cast(( select MONTH(GETDATE())) as varchar(50))+
            +'-' +cast((select DAY(GETDATE())) as varchar(50))))
    return @nb
end
go

-- get number of orders in a specific date
|create function NB_OFORDER2(@date_0 date)
returns integer
BEGIN
    DECLARE @nb integer
    SET @NB=(select count(*) from ORDER_INFO where Odate=@date_0)
    RETURN @NB
end
go
```

```
-- get number of products in a specific depot
create function NB_OFPRODUCT(@did int)
returns int
BEGIN
declare @nb int
set @nb=(select count(*) from products where did=@did AND QUANTITY!=0)
return @nb
end
go

-- get number of orders in a specific customer
create function NB_OFORDER(@cid int)
returns int
BEGIN
declare @nb int
set @nb=(select count(*) from order_INFO where cid=@cid)
return @nb
end
go
```

## Creating Triggers:

```
-- we can't insert 2 categories having the same name
    --- insert ---

create trigger CATEGORY_I on Category for insert
as
begin
declare @c int
set @c=(select count(*) from category)

if ((select count(*) from category t1,category t2 where t1.cat_name = t2.cat_name)!=@c)
begin
    rollback
end
end
go
    --- update ---

create trigger CATEGORY_U on category for update
as
begin
declare @c int
set @c=(select count(*) from category)
if ((select count(*) from category t1,category t2 where t1.cat_name = t2.cat_name)!=@c)
    rollback
end
go

    --- insert ---

create trigger SUPPLY_I on Supply for insert
as
begin
declare crs_SUPPLY_I cursor FOR select * from inserted
declare @pid int, @sid int, @DATE_S date, @qt decimal(8,0)

open crs_SUPPLY_I
fetch next from crs_SUPPLY_I into @SID,@PID,@DATE_S,@qt
while @@fetch_status=0
BEGIN
    update Products set quantity = quantity + @qt WHERE pid = @PID

    fetch next from crs_SUPPLY_I into @SID,@PID,@DATE_S,@qt
end
CLOSE crs_SUPPLY_I
deallocate crs_SUPPLY_I
end
go
```

Triggers on Category Table

Triggers on Supply Table

```

go
    --- delete ---

:create trigger SUPPLY_D on Supply for delete
as
begin
    declare crs_SUPPLY_D cursor FOR select * from deleted

declare @PID int, @SID int, @DATE_S date, @qt decimal(8,0)
open crs_SUPPLY_D
fetch next from crs_SUPPLY_D into @SID,@PID,@DATE_S,@qt
while @@fetch_status=0
BEGIN
    update Products set quantity = quantity - @qt WHERE pid = @PID
    fetch next from crs_SUPPLY_D into @SID,@PID,@DATE_S,@qt
end
CLOSE crs_SUPPLY_D
deallocate crs_SUPPLY_D
end
go

    --- update ---

:create trigger SUPPLY_U on SUPPLY for update
as
begin
    declare crs_SUPPLY_U1 cursor FOR select * from deleted
    declare crs_SUPPLY_U cursor FOR select * from inserted

declare @pid int, @sid int, @DATE_S date, @qt decimal(8,0)
declare @pid1 int, @sid1 int, @DATE_S1 date , @qt1 decimal(8,0)

open crs_SUPPLY_U
fetch next from crs_SUPPLY_U into @SID,@PID,@DATE_S,@qt
open crs_SUPPLY_U1
fetch next from crs_SUPPLY_U1 into @SID1,@PID1,@DATE_S1,@qt1
while @@fetch_status=0
BEGIN

    update Products set quantity = quantity - @qt1 where pid = @pid1
    update Products set quantity = quantity + @qt where pid = @pid

    fetch next from crs_SUPPLY_U into @SID,@PID,@DATE_S,@qt
    fetch next from crs_SUPPLY_U1 into @SID1,@PID1,@DATE_S1,@qt1
end
CLOSE crs_SUPPLY_U
deallocate crs_SUPPLY_U

CLOSE crs_SUPPLY_U1
deallocate crs_SUPPLY_U1
end
go

```

Trigger on Contain Table

```
--- insert ---

create trigger CONTAIN_I on Contain for insert
as
begin
    if exists(select 1 from products t1,inserted t2 where t1.pid=t2.pid and t1.quantity<t2.quantity)
        rollback

    declare crs_CONTAIN_I cursor FOR select * from inserted
    declare @OID int, @PID int, @qt decimal(8,0), @q decimal(8,0), @i CHAR(1), @pr int

    open crs_CONTAIN_I
    fetch next from crs_CONTAIN_I into @PID,@OID, @qt
    while @@fetch_status=0
        BEGIN
            set @pr = (select price from Products where pid=@PID)

            update Products set quantity = quantity-@qt where pid=@PID
            update Order_Info set TOTAL_AMOUNT = TOTAL_AMOUNT + @qt*@pr where oid=@OID

            fetch next from crs_CONTAIN_I into @PID,@OID,@qt
        END
    CLOSE crs_CONTAIN_I
    deallocate crs_CONTAIN_I
END
go
```

## Creating Views:

```
CREATE VIEW out_of_stock
AS select * from Products WHERE quantity=0
go

-----
CREATE VIEW on_stock
AS select * from Products WHERE quantity>0
go

-----
CREATE VIEW DIS
as select *
from discount
where (cast((select YEAR(GETDATE())) as varchar(50))+ '-' +cast(( select MONTH(GETDATE()) as varchar(50))
+ '-' +cast((select DAY(GETDATE())) as varchar(50)) between start_date and end_date)
        or
        (END_DATE IS NULL and START_DATE is NULL)
go

-----
CREATE VIEW ORDERTODAY
as select oid,total_amount
from order_info
where cast((select YEAR(GETDATE()) as varchar(50))+ '-' +cast(( select MONTH(GETDATE()) as varchar(50))
+ '-' +cast((select DAY(GETDATE())) as varchar(50))=odate
go
```

## Creating Rest of the Procedures:

```
-- delete all out dated discounts

|create procedure DELETE_DIS
AS
|begin
    declare @YY VARCHAR(5),@MM VARCHAR(5),@DD VARCHAR(5), @date DATE, @DIS_ID INT

    SET @YY=(select YEAR(GETDATE()))
    SET @MM=( select MONTH(GETDATE()))
    SET @DD=(select DAY(GETDATE()))
    set @date= @YY+'-'+@MM+'-'+@DD

    | declare crs CURSOR FOR select DIS_ID
        FROM DISCOUNT
        WHERE END_DATE < @date

        open crs
        fetch next from crs into @DIS_ID
        | while @@fetch_status=0
        | BEGIN
            UPDATE ORDER_INFO SET DIS_ID=NULL WHERE DIS_ID=@DIS_ID
            delete from DISCOUNT where DIS_ID=@DIS_ID
            fetch next from crs into @DIS_ID
        END
        CLOSE crs
        deallocate crs
    end
go
```

# Conclusion

---

This is the modern age, this system will allow customers to place orders without even visiting the shop, being able to buy anything any time, It became more enjoyable and easier than the real-world shopping.