CAN THO UNIVERSITY

COLLEGE OF INFORMATION AND COMMUNICATION TECHNOLOGY



**GROUP PROJECT**

# TOWASHOP PROJECT

### **Instructor: Group Members:**

Ph.D. Trần Công Án B2005853 – Nguyễn Thanh Phát B2005839 – Huỳnh Phi Hồng

## **Can Tho, 11/2022**

**Table of Contents**

[TOWASHOP PROJECT 1](#_Toc120478083)

[**Instructor:**  **Group Members:** 1](#_Toc120478084)

[**Can Tho, 11/2022** 1](#_Toc120478085)

[I. Data Design 4](#_Toc120478086)

[1. Categories 4](#_Toc120478087)

[ Code 4](#_Toc120478088)

[2. Products 5](#_Toc120478089)

[ Code 5](#_Toc120478090)

[3. Product\_Categories 6](#_Toc120478091)

[ Code 6](#_Toc120478092)

[4. Suppliers 6](#_Toc120478093)

[ Code 7](#_Toc120478094)

[5. Product\_Suppliers 7](#_Toc120478095)

[ Code 7](#_Toc120478096)

[6. Employees 8](#_Toc120478097)

[ Code 8](#_Toc120478098)

[7. Accounts 9](#_Toc120478099)

[ Code 9](#_Toc120478100)

[8. Employee\_Account 9](#_Toc120478101)

[ Code 10](#_Toc120478102)

[9. Roleviews 10](#_Toc120478103)

[ Code 10](#_Toc120478104)

[10. Account\_Roleviews 10](#_Toc120478105)

[ Code 10](#_Toc120478106)

[11. Orders 11](#_Toc120478107)

[ Code 11](#_Toc120478108)

[12. Order\_details 12](#_Toc120478109)

[ Code 12](#_Toc120478110)

[13. Supply\_Stocks 13](#_Toc120478111)

[ Code 13](#_Toc120478112)

[14. Customer 13](#_Toc120478113)

[ Code 14](#_Toc120478114)

[15. Customer\_Orders 14](#_Toc120478115)

[ Code 14](#_Toc120478116)

[II. List of stored procedures 14](#_Toc120478117)

[1. spCategory\_Delete 14](#_Toc120478118)

[2. spCategory\_GetAll 15](#_Toc120478119)

[3. spCategory\_GetByValue 15](#_Toc120478120)

[4. spCategory\_Insert 15](#_Toc120478121)

[5. spCategory\_Update 15](#_Toc120478122)

[6. spProduct\_Delete 15](#_Toc120478123)

[7. spProduct\_GetAll 15](#_Toc120478124)

[8. spProduct\_GetByValue 16](#_Toc120478125)

[9. spProduct\_Insert 16](#_Toc120478126)

[10. spProduct\_Update 17](#_Toc120478127)

[11. spSupplier\_Insert 18](#_Toc120478128)

[12. spSupplier\_GetAll 18](#_Toc120478129)

[13. spSupplier\_Update 18](#_Toc120478130)

[14. spSupplier\_Delete 19](#_Toc120478131)

[15. spSupplier\_GetByValue 19](#_Toc120478132)

[16. spCustomer\_Insert 19](#_Toc120478133)

[17. spCustomer\_GetAll 20](#_Toc120478134)

[18. spCustomer\_Update 20](#_Toc120478135)

[19. spCustomer\_Delete 20](#_Toc120478136)

[20. spCustomer\_GetByValue 20](#_Toc120478137)

[21. spOrder\_Insert 21](#_Toc120478138)

[22. spOrder\_Update 21](#_Toc120478139)

[23. spOrder\_GetAll 22](#_Toc120478140)

[24. spOrder\_GetByValue 22](#_Toc120478141)

[25. spProduct\_UpdateProductStock 22](#_Toc120478142)

[26. spProduct\_UpdateProductOrder 22](#_Toc120478143)

[27. spOrderDetails\_Insert 23](#_Toc120478144)

[28. spOrderDetails\_Update 23](#_Toc120478145)

[29. spOrderDetails\_GetByID 23](#_Toc120478146)

[30. spOrderDetail\_GetByID 24](#_Toc120478147)

[31. spOrderDetails\_Delete 24](#_Toc120478148)

[32. spStockOrder\_Insert 24](#_Toc120478149)

[33. spStockOrder\_Update 24](#_Toc120478150)

[34. spStockOrder\_GetAll 24](#_Toc120478151)

[35. spStockOrder\_GetByValue 24](#_Toc120478152)

[36. spCustomerOrder\_Insert 25](#_Toc120478153)

[37. spCustomerOrder\_Update 25](#_Toc120478154)

[38. spCustomerOrder\_GetAll 25](#_Toc120478155)

[39. spCustomerOrder\_GetByValue 25](#_Toc120478156)

[40. spAccount\_GetByUID 26](#_Toc120478157)

[41. spAccount\_GetByUsername 26](#_Toc120478158)

[42. spAccountRole\_GetAll 26](#_Toc120478159)

[43. spEmployee\_Delete 26](#_Toc120478160)

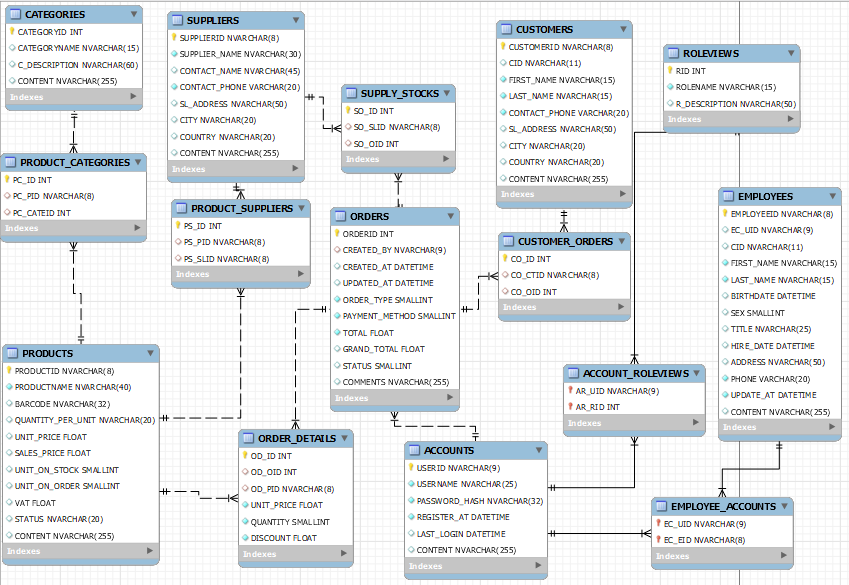
[44. spEmployee\_GetAll 27](#_Toc120478161)

[45. spEmployee\_GetByValue 27](#_Toc120478162)

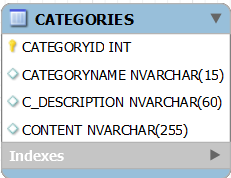
[46. spEmployee\_Insert 27](#_Toc120478163)

[47. spEmployee\_Update 28](#_Toc120478164)

# Data Design



## Categories



|  |  |
| --- | --- |
| **CategoryID** | The unique id to identify the category. |
| **Categoryname** | The category title. |
| **C\_Description** | The column used to store the category details. |
| **Content** | The column for additional details if available. |

### Code

CREATE TABLE CATEGORIES(

CATEGORYID INT AUTO\_INCREMENT,

CATEGORYNAME NVARCHAR(15) UNIQUE,

C\_DESCRIPTION NVARCHAR(60) NULL DEFAULT NULL,

CONTENT NVARCHAR(255) NULL DEFAULT NULL,

CONSTRAINT PK\_CATEGOIRES

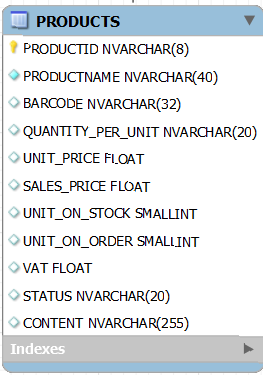
PRIMARY KEY (CATEGORYID),

CONSTRAINT CK\_UNIQUE\_NAME

UNIQUE(CATEGORYNAME)

);

## Products



|  |  |
| --- | --- |
| **ProductID** | The unique id to identify the product. |
| **Productname** | The product title to be displayed on the Inventory. |
| **Barcode** | The column storing the barcode value |
| **Quantity\_per\_unit** | It shows the measurement or quantity per unit of sale |
| **Unit\_price** | It stores the original (without discounts and taxes) listed price of the product per unit |
| **Sales\_price** | It stores the current sale price per unit |
| **Unit\_on\_stock** | Shows the number of items currently in storage |
| **Unit\_on\_order** | Shows the number of items on order |
| **Vat** | The VAT applied to the products |
| **Status** | The current sales status of the product, it could be in sales or not |
| **Content** | The column used to store the additional details of the product. |

### Code

CREATE TABLE PRODUCTS(

PRODUCTID NVARCHAR(8),

PRODUCTNAME NVARCHAR(40) NOT NULL,

BARCODE NVARCHAR(32) NULL DEFAULT NULL,

QUANTITY\_PER\_UNIT NVARCHAR(20) NULL,

UNIT\_PRICE FLOAT DEFAULT (0),

SALES\_PRICE FLOAT DEFAULT (0),

UNIT\_ON\_STOCK SMALLINT DEFAULT(0),

UNIT\_ON\_ORDER SMALLINT DEFAULT(0),

VAT FLOAT DEFAULT (0),

STATUS NVARCHAR(20) NULL,

CONTENT NVARCHAR(255) NULL DEFAULT NULL,

CONSTRAINT PK\_PRODUCTS

PRIMARY KEY (PRODUCTID),

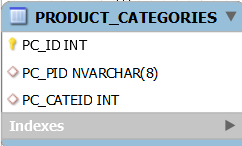
CONSTRAINT CK\_PRODUCTS

CHECK (UNIT\_PRICE>=0),

CHECK (SALES\_PRICE>=0)

);

## Product\_Categories



|  |  |
| --- | --- |
| **PC\_ID** | The unique id to identify the product – categories record. |
| **PC\_PID** | The product id to identify the product.  It references ProductID of Products table |
| **PC\_CATEID INT** | The category id to identify the category.  It references CategoryID of Categories table |

### Code

CREATE TABLE PRODUCT\_CATEGORIES(

PC\_ID INT AUTO\_INCREMENT,

PC\_PID NVARCHAR(8) UNIQUE,

PC\_CATEID INT DEFAULT(0)NULL,

CONSTRAINT PK\_PRODUCT\_CATEGORIES

PRIMARY KEY (PC\_ID),

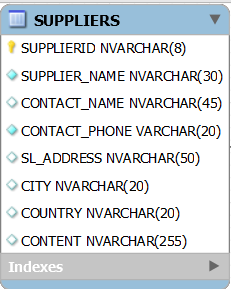
CONSTRAINT FK\_PRODUCT\_CATEGORIES

FOREIGN KEY (PC\_PID) REFERENCES PRODUCTS(PRODUCTID) ON DELETE CASCADE,

FOREIGN KEY(PC\_CATEID) REFERENCES CATEGORIES(CATEGORYID) ON DELETE CASCADE

);

## Suppliers



|  |  |
| --- | --- |
| **SupplierID** | The unique id to identify the supplier |
| **Supplier\_name** | The name of supplier to be displayed on the application |
| **Contact\_name** | The name of person in contact on the supplier side. |
| **Contact\_phone** | The contact phone number of the supplier’s enterprise |
| **SL\_address** | The detailed address of the supplier enterprise. |
| **City** | The city of the supplier |
| **Country** | The country of the supplier |
| **Content** | Additional information regarding the supplier |

### Code

CREATE TABLE SUPPLIERS(

SUPPLIERID NVARCHAR(8),

SUPPLIER\_NAME NVARCHAR(30) NOT NULL,

CONTACT\_NAME NVARCHAR(45) NULL,

CONTACT\_PHONE VARCHAR(20) NOT NULL,

SL\_ADDRESS NVARCHAR(50) NULL DEFAULT NULL,

CITY NVARCHAR(20) NULL DEFAULT NULL,

COUNTRY NVARCHAR(20) NULL DEFAULT NULL,

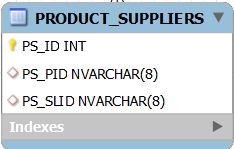
CONTENT NVARCHAR(255) NULL DEFAULT NULL,

CONSTRAINT PK\_SUPPLIERS

PRIMARY KEY (SUPPLIERID)

);

## Product\_Suppliers



|  |  |
| --- | --- |
| **PS\_ID** | The unique id to identify the product – supplier record. |
| **PC\_PID** | The product id to identify the product.  It references ProductID of Products table |
| **PC\_CATEID INT** | The supplier id to identify the supplier.  It references SupplierID of Suppliers table |

### Code

CREATE TABLE PRODUCT\_SUPPLIERS(

PS\_ID INT AUTO\_INCREMENT,

PS\_PID NVARCHAR(8) UNIQUE,

PS\_SLID NVARCHAR(8) NULL,

CONSTRAINT PK\_PRODUCT\_SUPPLIERS

PRIMARY KEY(PS\_ID),

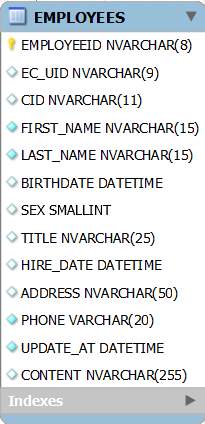
CONSTRAINT FK\_PRODUCT\_SUPPLIERS

FOREIGN KEY (PS\_PID) REFERENCES PRODUCTS(PRODUCTID) ON DELETE CASCADE,

FOREIGN KEY (PS\_SLID) REFERENCES SUPPLIERS(SUPPLIERID) ON DELETE CASCADE

);

## Employees



|  |  |
| --- | --- |
| **EmployeeID** | The unique id to identify the Employee. |
| **EC\_UID** | The username of the employee on the application. |
| **CID** | The citizen identification code of the employee. |
| **First\_name** | The first name of the employee. |
| **Last\_name** | The last name of the employee. |
| **Birthdate** | The birthdate of the employee. |
| **Sex** | The gender of the employee. |
| **Title** | The title of the employee. |
| **Hire\_date** | The date when the employee was hired. |
| **Address** | The Address of the employee. |
| **Phone** | The phone number of the employee. |
| **Update\_at** | The last time when the employee’s information was changed. |
| **Content** | Additional information regarding the employee. |

### Code

CREATE TABLE EMPLOYEES(

EMPLOYEEID NVARCHAR(8),

EC\_UID NVARCHAR(9) NULL,

CID NVARCHAR(11) NULL,

FIRST\_NAME NVARCHAR(15) NOT NULL,

LAST\_NAME NVARCHAR(15) NOT NULL,

BIRTHDATE DATETIME NULL,

SEX SMALLINT NULL DEFAULT '0',

TITLE NVARCHAR(25) NULL,

HIRE\_DATE DATETIME NULL,

ADDRESS NVARCHAR(50) NULL,

PHONE VARCHAR(20) NOT NULL,

UPDATE\_AT DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP,

CONTENT NVARCHAR(255) NULL DEFAULT NULL,

CONSTRAINT PK\_EMPLOYEES

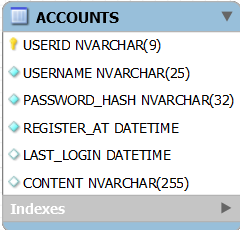
PRIMARY KEY (EMPLOYEEID),

CONSTRAINT CK\_UNIQUE\_CID

UNIQUE(CID)

);

## Accounts



|  |  |
| --- | --- |
| **UserID** | The unique id to identify the user. |
| **Username** | The username of the user on the application. |
| **Password\_Hash** | The password hash of the user generated by the appropriate algorithm |
| **Register\_at** | The time when the account was registered |
| **Last\_login** | The last time when the account was logged in. |
| **Content** | Additional information regarding the account. |

### Code

CREATE TABLE ACCOUNTS(

USERID NVARCHAR(9),

USERNAME NVARCHAR(25) NOT NULL,

PASSWORD\_HASH NVARCHAR(32) NOT NULL,

REGISTER\_AT DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP,

LAST\_LOGIN DATETIME NULL,

CONTENT NVARCHAR(255) NULL DEFAULT NULL,

CONSTRAINT PK\_ACCOUNTS

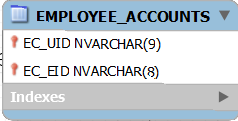
PRIMARY KEY (USERID),

CONSTRAINT CK\_UNIQUE\_USERNAME

UNIQUE(USERNAME)

);

## Employee\_Account



|  |  |
| --- | --- |
| **EC\_UID** | The username of the employee on the application.  It references UserID of Accounts table |
| **EC\_EID** | The unique id to identify the Employee.  It references EmployeeID of Employees table |

### Code

CREATE TABLE EMPLOYEE\_ACCOUNTS(

EC\_UID NVARCHAR(9) NOT NULL,

EC\_EID NVARCHAR(8) NOT NULL,

CONSTRAINT PK\_EMPLOYEE\_ACCOUNTS

PRIMARY KEY(EC\_UID,EC\_EID),

CONSTRAINT FK\_EMPLOYEE\_ACCOUNTS

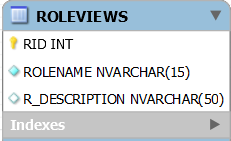
FOREIGN KEY (EC\_UID) REFERENCES ACCOUNTS(USERID),

FOREIGN KEY (EC\_EID) REFERENCES EMPLOYEES (EMPLOYEEID)

ON DELETE CASCADE

);

## Roleviews



|  |  |
| --- | --- |
| **RID** | The unique id to identify the roleview within the application. |
| **Rolename** | The name of the role within the application. |
| **R\_description** | Stores detail information of the role. |

### Code

CREATE TABLE ROLEVIEWS(

RID INT AUTO\_INCREMENT,

ROLENAME NVARCHAR(15) NOT NULL,

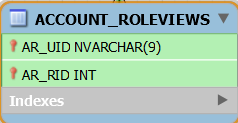
R\_DESCRIPTION NVARCHAR(50) NULL DEFAULT NULL,

CONSTRAINT PK\_ROLEVIEWS

PRIMARY KEY (RID)

);

## Account\_Roleviews



|  |  |
| --- | --- |
| **AR\_UID** | The unique id to identify the user.  It references UserID of Accounts table |
| **AR\_RID** | The unique id to identify the roleview.  It references RID of Roleviews table |

### Code

CREATE TABLE ACCOUNT\_ROLEVIEWS(

AR\_UID NVARCHAR(9) NOT NULL,

AR\_RID INT NOT NULL,

CONSTRAINT PK\_ACCOUNT\_ROLES

PRIMARY KEY (AR\_UID,AR\_RID),

CONSTRAINT FK\_ACCOUNT\_ROLES

FOREIGN KEY (AR\_UID) REFERENCES ACCOUNTS(USERID)

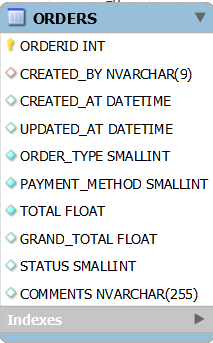
ON DELETE CASCADE,

FOREIGN KEY (AR\_RID) REFERENCES ROLEVIEWS(RID)

ON DELETE CASCADE

);

## Orders



|  |  |
| --- | --- |
| **OrderID** | The unique id to identify the order. |
| **Created\_by** | The account that created the order.  It references UserID of Accounts table |
| **Created\_at** | The time that the order was created. |
| **Updated\_at** | The last time that the order was updated. |
| **Order\_type** | The type of the order. 0 is retail order, 1 is customer order, 2 is stock purchase |
| **Payment\_method** | Payment method used for the order. 0 is cash, and 1 is using credit card |
| **Total** | The total original price needs to be paid of the order |
| **Grand\_total** | The total price accounted discounts and taxes |
| **Status** | The status of the order. 0 is new, 1 is paid, 2 is delivering |
| **Comment** | Additional notification or comments for the order |

### Code

CREATE TABLE ORDERS(

ORDERID INT AUTO\_INCREMENT,

CREATED\_BY NVARCHAR(9) NULL DEFAULT NULL,

CREATED\_AT DATETIME NULL DEFAULT CURRENT\_TIMESTAMP,

UPDATED\_AT DATETIME NULL DEFAULT NULL,

ORDER\_TYPE SMALLINT NOT NULL DEFAULT (0),

PAYMENT\_METHOD SMALLINT NOT NULL DEFAULT (0), -- 0 CASH - 1 CREDIT

TOTAL FLOAT NOT NULL DEFAULT 0,

GRAND\_TOTAL FLOAT NULL DEFAULT (0),

STATUS SMALLINT NULL DEFAULT (0), -- NEW

COMMENTS NVARCHAR(255) NULL DEFAULT NULL,

CONSTRAINT PK\_ORDERS

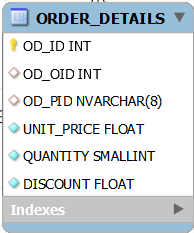
PRIMARY KEY (ORDERID),

CONSTRAINT FK\_ORDERS

FOREIGN KEY(CREATED\_BY) REFERENCES ACCOUNTS(USERID)

);

## Order\_details



|  |  |
| --- | --- |
| **OD\_ID** | The unique id to identify the order detail. |
| **OD\_OID** | The unique id to identify the order.  It references OrderID of Orders table |
| **OD\_PID** | The unique id to identify the product.  It references ProductID of Products table |
| **Unit\_price** | The total price of the product |
| **Quantity** | The quantity that the order requested for the product |
| **Discount** | The percent of discount applied to the product for that specified order |

### Code

CREATE TABLE ORDER\_DETAILS(

OD\_ID INT AUTO\_INCREMENT,

OD\_OID INT,

OD\_PID NVARCHAR(8),

UNIT\_PRICE FLOAT NOT NULL DEFAULT (0),

QUANTITY SMALLINT NOT NULL DEFAULT (1),

DISCOUNT FLOAT NOT NULL DEFAULT(0),

CONSTRAINT PK\_ORDER\_DETAILS

PRIMARY KEY (OD\_ID),

CONSTRAINT FK\_ORDER\_DETAILS

FOREIGN KEY (OD\_OID) REFERENCES ORDERS(ORDERID) ON DELETE CASCADE,

FOREIGN KEY (OD\_PID) REFERENCES PRODUCTS(PRODUCTID),

CONSTRAINT CK\_DISCOUNT

CHECK (DISCOUNT>=0 AND DISCOUNT<=1),

CONSTRAINT CK\_QUANTITY

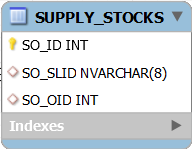
CHECK(QUANTITY>0),

CONSTRAINT CK\_UNIT\_PRICE

CHECK (UNIT\_PRICE>=0)

);

## Supply\_Stocks



|  |  |
| --- | --- |
| **SO\_ID** | The unique id to identify the stock order. |
| **SO\_SLID** | The unique id to identify the supplier.  It references SupplierID of Suppliers table |
| **SO\_OID** | The unique id to identify the order.  It references OrderID of Orders table |

### Code

CREATE TABLE SUPPLY\_STOCKS(

SO\_ID INT AUTO\_INCREMENT,

SO\_SLID NVARCHAR(8),

SO\_OID INT,

CONSTRAINT PK\_ORDER\_SUPPLIERS

PRIMARY KEY (SO\_ID),

CONSTRAINT CK\_ORDER\_SUPPLIERS

UNIQUE (SO\_SLID,SO\_OID),

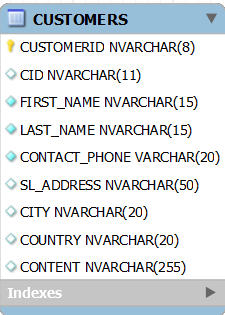
CONSTRAINT FK\_ORDER\_SUPPLIERS

FOREIGN KEY (SO\_SLID) REFERENCES SUPPLIERS(SUPPLIERID),

FOREIGN KEY (SO\_OID) REFERENCES ORDERS(ORDERID)

);

## Customer



|  |  |
| --- | --- |
| **CustomerID** | The unique id to identify the customer. |
| **CID** | The citizen identification code of the customer. |
| **First\_name** | The first name of the customer. |
| **Last\_name** | The last name of the customer. |
| **Contact\_phone** | The contact phone of the customer’s enterprise. |
| **SL\_address** | The detailed address of the customer’s enterprise. |
| **City** | The city of the customer |
| **Country** | The country of the customer |
| **Content** | Additional information regarding the customer |

### Code

CREATE TABLE CUSTOMERS(

CUSTOMERID NVARCHAR(8), -- CT1000 ~

CID NVARCHAR(11) NULL,

FIRST\_NAME NVARCHAR(15) NOT NULL,

LAST\_NAME NVARCHAR(15) NOT NULL,

CONTACT\_PHONE VARCHAR(20) NOT NULL,

SL\_ADDRESS NVARCHAR(50) NULL DEFAULT NULL,

CITY NVARCHAR(20) NULL DEFAULT NULL,

COUNTRY NVARCHAR(20) NULL DEFAULT NULL,

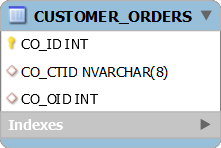
CONTENT NVARCHAR(255) NULL DEFAULT NULL,

CONSTRAINT PK\_CUSTOMERS

PRIMARY KEY (CUSTOMERID)

);

## Customer\_Orders



|  |  |
| --- | --- |
| **CO\_ID** | The unique id to identify the customer order. |
| **CO\_CTID** | The unique id to identify the customer.  It references CustomerID of Customers table |
| **CO\_OID** | The unique id to identify the order.  It references OrderID of Orders table |

### Code

CREATE TABLE CUSTOMER\_ORDERS(

CO\_ID INT AUTO\_INCREMENT,

CO\_CTID NVARCHAR(8),

CO\_OID INT,

CONSTRAINT PK\_CUSTOMER\_ORDERS

PRIMARY KEY (CO\_ID),

CONSTRAINT CK\_CUSTOMER\_ORDERS

UNIQUE (CO\_CTID,CO\_OID),

CONSTRAINT FK\_CUSTOMER\_ORDERS

FOREIGN KEY (CO\_CTID) REFERENCES CUSTOMERS(CUSTOMERID) ,

FOREIGN KEY (CO\_OID) REFERENCES ORDERS(ORDERID)

);

# List of stored procedures

## spCategory\_Delete

DELIMITER $$

**CREATE** **PROCEDURE** spCategory\_Delete (**IN** CID **INT**)

**BEGIN**

**DELETE** **FROM** CATEGORIES

**WHERE** CATEGORYID = CID;

**END** $$

## spCategory\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spCategory\_GetAll ()

**BEGIN**

**SELECT** \* **FROM** CATEGORIES ;

**END** $$

## spCategory\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spCategory\_GetByValue (**IN** **VALUE** **NVARCHAR**(50))

**BEGIN**

**SELECT** **DISTINCT** C.\* **FROM** CATEGORIES C

**LEFT** **JOIN** PRODUCT\_CATEGORIES PC **ON** PC.PC\_CATEID = C.CATEGORYID

**LEFT** **JOIN** PRODUCTS P **ON** PC.PC\_PID = P.PRODUCTID

**WHERE** **cast**(C.CATEGORYID **as** **char**) **LIKE** **VALUE**

**OR** C.CATEGORYNAME **LIKE** **VALUE**

**OR** C.C\_DESCRIPTION **LIKE** **VALUE**

**OR** P.PRODUCTID **LIKE** **VALUE**

**OR** P.BARCODE **LIKE** **VALUE**

**OR** P.PRODUCTNAME **LIKE** **VALUE**;

**END** $$

## spCategory\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spCategory\_Insert (**IN** CATEGORYNAME **NVARCHAR**(15), **IN** C\_DESCRIPTION **NVARCHAR**(60), **IN** CONTENT **NVARCHAR**(255))

**BEGIN**

**INSERT** **INTO** CATEGORIES (CATEGORYNAME,C\_DESCRIPTION,CONTENT)

**VALUES** (CATEGORYNAME,C\_DESCRIPTION,CONTENT);

**END** $$

## spCategory\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spCategory\_Update (**IN** CID **INT**, **IN** CATEGORYNAME **NVARCHAR**(15), **IN** C\_DESCRIPTION **NVARCHAR**(60), **IN** CONTENT **NVARCHAR**(255))

**BEGIN**

**UPDATE** CATEGORIES

**SET**

CATEGORIES.CATEGORYNAME = CATEGORYNAME,

CATEGORIES.C\_DESCRIPTION = C\_DESCRIPTION,

CATEGORIES.CONTENT = CONTENT

**WHERE** CATEGORIES.CATEGORYID = CID;

**END** $$

## spProduct\_Delete

DELIMITER $$

**CREATE** **PROCEDURE** spProduct\_Delete (**IN** PID **NVARCHAR** (8))

**BEGIN**

**DELETE** **FROM** PRODUCTS **WHERE** PRODUCTID = PID;

**END** $$

## spProduct\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spProduct\_GetAll ()

**BEGIN**

**SELECT** P.\*, PC\_CATEID,PS\_SLID **FROM** PRODUCTS P

**LEFT** **JOIN** PRODUCT\_SUPPLIERS PS **ON** P.PRODUCTID = PS.PS\_PID

**LEFT** **JOIN** PRODUCT\_CATEGORIES PC **ON** P.PRODUCTID = PC.PC\_PID;

**END** $$

## spProduct\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spProduct\_GetByValue (**IN** **VALUE** **NVARCHAR**(30))

**BEGIN**

**SELECT** **DISTINCT** P.\*, PC.PC\_CATEID, PS.PS\_SLID **FROM** PRODUCTS P

**LEFT** **JOIN** PRODUCT\_SUPPLIERS PS **ON** P.PRODUCTID = PS.PS\_PID

**LEFT** **JOIN** PRODUCT\_CATEGORIES PC **ON** P.PRODUCTID = PC.PC\_PID

**LEFT** **JOIN** SUPPLIERS S **ON** S.SUPPLIERID = PS.PS\_SLID

**LEFT** **JOIN** CATEGORIES C **ON** C.CATEGORYID = PC.PC\_CATEID

**WHERE** P.PRODUCTID **LIKE** **VALUE**

**OR** P.BARCODE = **VALUE**

**OR** P.PRODUCTNAME **LIKE** **VALUE**

**OR** P.STATUS **LIKE** **VALUE**

**OR** P.CONTENT **LIKE** **VALUE**

**OR** P.QUANTITY\_PER\_UNIT **LIKE** **VALUE**

**OR** S.SUPPLIERID **LIKE** **VALUE**

**OR** S.SUPPLIER\_NAME **LIKE** **VALUE**

**OR** S.CONTACT\_NAME **LIKE** **VALUE**

**OR** S.CONTACT\_PHONE **LIKE** **VALUE**

**OR** S.CITY **LIKE** **VALUE**

**OR** S.COUNTRY **LIKE** **VALUE**

**OR** C.CATEGORYNAME **LIKE** **VALUE**;

**END** $$

## spProduct\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spProduct\_Insert (

**IN** PRODUCTNAME **NVARCHAR**(40),

**IN** BARCODE **NVARCHAR**(32),

**IN** QUANTITY\_PER\_UNIT **NVARCHAR**(20),

**IN** UNIT\_PRICE **FLOAT**,

**IN** SALES\_PRICE **FLOAT**,

**IN** UNIT\_ON\_STOCK **SMALLINT**,

**IN** UNIT\_ON\_ORDER **SMALLINT**,

**IN** VAT **FLOAT**,

**IN** STATUS **NVARCHAR**(20),

**IN** CONTENT **NVARCHAR**(255),

**IN** PC\_CATEID **INT**,

**IN** PS\_SLID **NVARCHAR**(8)

)

**BEGIN**

-- initialize variables

**DECLARE** NEWPID **nvarchar**(8);

**DECLARE** CURRENTPID **nvarchar**(8);

**DECLARE** **NEWID** **INT**;

-- set default values

**IF** UNIT\_PRICE **IS** **NULL** **THEN** **SET** UNIT\_PRICE = 0.0; **END** **IF**;

**IF** SALES\_PRICE **IS** **NULL** **THEN** **SET** SALES\_PRICE = UNIT\_PRICE; **END** **IF**;

**IF** UNIT\_ON\_STOCK **IS** **NULL** **THEN** **SET** UNIT\_ON\_STOCK = 0; **END** **IF**;

**IF** UNIT\_ON\_ORDER **IS** **NULL** **THEN** **SET** UNIT\_ON\_ORDER = 0; **END** **IF**;

**IF** VAT **IS** **NULL** **THEN** **SET** VAT = 0; **END** **IF**;

**IF** PC\_CATEID **IS** **NULL** **THEN** **SET** PC\_CATEID = 0; **END** **IF**;

**IF** NEWPID **IS** **NULL** **THEN** **SET** NEWPID = 'P1000';**END** **IF**;

-- CHECK TABLE IS CONTAIN ROWS

**IF** **EXISTS** (**SELECT** 1 **FROM** PRODUCTS) **THEN**

**BEGIN**

-- TAKE THE LARGEST VALUE OF ID AND PLUS ONE IN FORMAT

**SET** CURRENTPID = (**Select** **Max**(PRODUCTID) **from** PRODUCTS P);

**SET** **NEWID** = (**SELECT** **Cast**(**SUBSTRING**(CURRENTPID,2,**LENGTH**(CURRENTPID)) **AS** UNSIGNED)+1) ;

**SET** NEWPID = **Concat**('P',**CAST**(**NEWID** **as** **char**));

**END**;

**END** **IF**;

**INSERT** **INTO** PRODUCTS (PRODUCTID,PRODUCTNAME,BARCODE,

QUANTITY\_PER\_UNIT,UNIT\_PRICE,SALES\_PRICE,

UNIT\_ON\_STOCK,UNIT\_ON\_ORDER,VAT,STATUS,CONTENT)

**VALUES**(NEWPID,PRODUCTNAME,BARCODE,

QUANTITY\_PER\_UNIT,UNIT\_PRICE,SALES\_PRICE,

UNIT\_ON\_STOCK,UNIT\_ON\_ORDER,VAT,STATUS,CONTENT);

**INSERT** **INTO** PRODUCT\_CATEGORIES **VALUES** (NEWPID,PC\_CATEID);

**INSERT** **INTO** PRODUCT\_SUPPLIERS **VALUES** (NEWPID,PS\_SLID);

**END** $$

## spProduct\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spProduct\_Update (

**IN** PRODUCTID **NVARCHAR**(8),

**IN** PRODUCTNAME **NVARCHAR**(40),

**IN** BARCODE **NVARCHAR**(32),

**IN** QUANTITY\_PER\_UNIT **NVARCHAR**(20),

**IN** UNIT\_PRICE **FLOAT**,

**IN** SALES\_PRICE **FLOAT**,

**IN** UNIT\_ON\_STOCK **SMALLINT**,

**IN** UNIT\_ON\_ORDER **SMALLINT**,

**IN** VAT **FLOAT**,

**IN** STATUS **NVARCHAR**(20),

**IN** CONTENT **NVARCHAR**(255),

**IN** PC\_CATEID **INT**,

**IN** PS\_SLID **NVARCHAR**(8)

)

**BEGIN**

-- set default values

**IF** UNIT\_PRICE **IS** **NULL** **THEN** **SET** UNIT\_PRICE = 0.0; **END** **IF**;

**IF** SALES\_PRICE **IS** **NULL** **THEN** **SET** SALES\_PRICE = UNIT\_PRICE; **END** **IF**;

**IF** UNIT\_ON\_STOCK **IS** **NULL** **THEN** **SET** UNIT\_ON\_STOCK = 0; **END** **IF**;

**IF** UNIT\_ON\_ORDER **IS** **NULL** **THEN** **SET** UNIT\_ON\_ORDER = 0; **END** **IF**;

**IF** VAT **IS** **NULL** **THEN** **SET** VAT = 0; **END** **IF**;

**IF** PC\_CATEID **IS** **NULL** **THEN** **SET** PC\_CATEID = 0; **END** **IF**;

**UPDATE** PRODUCTS

**SET**

PRODUCTS.PRODUCTNAME =PRODUCTNAME,

PRODUCTS.BARCODE =BARCODE,

PRODUCTS.QUANTITY\_PER\_UNIT =QUANTITY\_PER\_UNIT,

PRODUCTS.UNIT\_PRICE= UNIT\_PRICE ,

PRODUCTS.SALES\_PRICE = SALES\_PRICE,

PRODUCTS.UNIT\_ON\_STOCK =UNIT\_ON\_STOCK,

PRODUCTS.UNIT\_ON\_ORDER= UNIT\_ON\_ORDER,

PRODUCTS.VAT=VAT ,

PRODUCTS.STATUS = STATUS,

PRODUCTS.CONTENT= CONTENT

**WHERE** PRODUCTS.PRODUCTID = PRODUCTID;

**UPDATE** PRODUCT\_CATEGORIES

**SET**

PRODUCT\_CATEGORIES.PC\_CATEID= PC\_CATEID

**WHERE**

PRODUCT\_CATEGORIES.PC\_PID = PRODUCTID;

**UPDATE** PRODUCT\_SUPPLIERS

**SET**

PRODUCT\_SUPPLIERS.PS\_SLID= PS\_SLID

**WHERE**

PRODUCT\_SUPPLIERS.PS\_PID = @PRODUCTID;

**END** $$

## spSupplier\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spSupplier\_Insert (

**IN** SUPPLIER\_NAME **NVARCHAR**(30),

**IN** CONTACT\_NAME **NVARCHAR**(45),

**IN** CONTACT\_PHONE **NVARCHAR**(20),

**IN** SL\_ADDRESS **NVARCHAR**(50),

**IN** CITY **NVARCHAR**(20),

**IN** COUNTRY **NVARCHAR**(20),

**IN** CONTENT **NVARCHAR**(255)

)

**BEGIN**

-- initialize variables

**DECLARE** NEWSLID **nvarchar**(8);

**DECLARE** CURRENTSLID **nvarchar**(8);

**DECLARE** **NEWID** **INT**;

-- set default values

**IF** NEWSLID **IS** **NULL** **THEN** **SET** NEWSLID = 'SL1000'; **END** **IF**;

-- CHECK TABLE IS CONTAIN ROWS

**IF** **EXISTS** (**SELECT** 1 **FROM** SUPPLIERS) **THEN**

**BEGIN**

-- TAKE THE LARGEST VALUE OF ID AND PLUS ONE IN FORMAT

**SET** CURRENTSLID = (**Select** **Max**(SUPPLIERID) **from** SUPPLIERS S);

**SET** **NEWID** = (**SELECT** **Cast**(**SUBSTRING**(CURRENTSLID,3,**LENGTH**(CURRENTSLID)) **AS** UNSIGNED)+1) ;

**SET** NEWSLID = **Concat**('SL',**CAST**(**NEWID** **as** **char**));

**END**;

**END** **IF**;

-- Insert statements for procedure here

**INSERT** **INTO** SUPPLIERS (SUPPLIERID,SUPPLIER\_NAME, CONTACT\_NAME, CONTACT\_PHONE, SL\_ADDRESS, CITY, COUNTRY , CONTENT)

**VALUES** (NEWSLID, SUPPLIER\_NAME, CONTACT\_NAME, CONTACT\_PHONE, SL\_ADDRESS, CITY, COUNTRY, CONTENT);

**END** $$

## spSupplier\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spSupplier\_GetAll ()

**BEGIN**

**SELECT** \* **FROM** SUPPLIERS ;

**END** $$

## spSupplier\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spSupplier\_Update (

**IN** SLID **NVARCHAR**(15), **IN** SUPPLIER\_NAME **NVARCHAR**(30), **IN** CONTACT\_NAME **NVARCHAR**(45), **IN** CONTACT\_PHONE **NVARCHAR**(20),

**IN** SL\_ADDRESS **NVARCHAR**(50), **IN** CITY **NVARCHAR**(20), **IN** COUNTRY **NVARCHAR**(20), **IN** CONTENT **NVARCHAR**(255)

)

**BEGIN**

**UPDATE** SUPPLIERS

**SET** SUPPLIERS.SUPPLIER\_NAME = SUPPLIER\_NAME,

SUPPLIERS.CONTACT\_NAME = CONTACT\_NAME,

SUPPLIERS.CONTACT\_PHONE = CONTACT\_PHONE,

SUPPLIERS.SL\_ADDRESS= SL\_ADDRESS,

SUPPLIERS.CITY = CITY,

SUPPLIERS.COUNTRY = COUNTRY,

SUPPLIERS.CONTENT = CONTENT

**WHERE** SUPPLIERS.SUPPLIERID =SLID;

**END** $$

## spSupplier\_Delete

DELIMITER $$

**CREATE** **PROCEDURE** spSupplier\_Delete (**IN** SLID **nvarchar**(15))

**BEGIN**

**DELETE** **FROM** SUPPLIERS

**WHERE** SUPPLIERID = SLID;

**END** $$

## spSupplier\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spSupplier\_GetByValue (**IN** SEARCHVALUE **nvarchar**(50))

**BEGIN**

**SELECT** SL.\* **FROM** SUPPLIERS SL

**LEFT** **JOIN** PRODUCT\_SUPPLIERS PS **ON** SL.SUPPLIERID = PS.PS\_SLID

**LEFT** **JOIN** PRODUCTS P **ON** P.PRODUCTID = PS.PS\_SLID

**WHERE** SL.SUPPLIER\_NAME **LIKE** SEARCHVALUE

**OR** SL.SUPPLIERID **LIKE** SEARCHVALUE

**OR** SL.CONTACT\_NAME **LIKE** SEARCHVALUE

**OR** SL.CONTACT\_PHONE **LIKE** SEARCHVALUE

**OR** SL.SL\_ADDRESS **LIKE** SEARCHVALUE

**OR** P.PRODUCTID **LIKE** SEARCHVALUE

**OR** P.PRODUCTNAME **LIKE** SEARCHVALUE

**ORDER** **BY** SL.SUPPLIERID;

**END** $$

## spCustomer\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spCustomer\_Insert (

**IN** CID **NVARCHAR**(11),

**IN** FIRST\_NAME **NVARCHAR**(15),

**IN** LAST\_NAME **NVARCHAR**(15),

**IN** CONTACT\_PHONE **VARCHAR**(20),

**IN** SL\_ADDRESS **NVARCHAR**(50),

**IN** CITY **NVARCHAR**(20),

**IN** COUNTRY **NVARCHAR**(20),

**IN** CONTENT **NVARCHAR**(255)

)

**BEGIN**

-- initialize variables

**DECLARE** NEWCTID **nvarchar**(8);

**DECLARE** CURRENTCTID **nvarchar**(8);

**DECLARE** **NEWID** **INT**;

-- set default values

**IF** NEWCTID **IS** **NULL** **THEN** **SET** NEWCTID = 'CT1000'; **END** **IF**;

-- CHECK TABLE IS CONTAIN ROWS

**IF** **EXISTS** (**SELECT** 1 **FROM** CUSTOMERS) **THEN**

**BEGIN**

-- TAKE THE LARGEST VALUE OF ID AND PLUS ONE IN FORMAT

**SET** CURRENTCTID = (**Select** **Max**(CUSTOMERID) **from** CUSTOMERS CT);

**SET** **NEWID** = (**SELECT** **Cast**(**SUBSTRING**(CURRENTCTID,3,**LENGTH**(CURRENTCTID)) **AS** UNSIGNED)+1) ;

**SET** NEWCTID = **Concat**('CT',**CAST**(**NEWID** **as** **char**));

**END**;

**END** **IF**;

-- Insert statements for procedure here

**INSERT** **INTO** CUSTOMERS (CUSTOMERID, CID, FIRST\_NAME, LAST\_NAME, CONTACT\_PHONE, SL\_ADDRESS, CITY, COUNTRY , CONTENT)

**VALUES** (NEWCTID, CID, FIRST\_NAME, LAST\_NAME, CONTACT\_PHONE, SL\_ADDRESS, CITY, COUNTRY, CONTENT);

**END** $$

## spCustomer\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spCustomer\_GetAll ()

**BEGIN**

**SELECT** \* **FROM** CUSTOMERS ;

**END** $$

## spCustomer\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spCustomer\_Update (

**IN** CTID **NVARCHAR**(8), **IN** CID **NVARCHAR**(11), **IN** FIRST\_NAME **NVARCHAR**(15),

**IN** LAST\_NAME **NVARCHAR**(15), **IN** CONTACT\_PHONE **NVARCHAR**(20), **IN** SL\_ADDRESS **NVARCHAR**(50),

**IN** CITY **NVARCHAR**(20), **IN** COUNTRY **NVARCHAR**(20), **IN** CONTENT **NVARCHAR**(255)

)

**BEGIN**

**UPDATE** CUSTOMERS

**SET** CUSTOMERS.CID = CID,

CUSTOMERS.FIRST\_NAME = FIRST\_NAME,

CUSTOMERS.LAST\_NAME = LAST\_NAME,

CUSTOMERS.CONTACT\_PHONE = CONTACT\_PHONE,

CUSTOMERS.SL\_ADDRESS= SL\_ADDRESS,

CUSTOMERS.CITY = CITY,

CUSTOMERS.COUNTRY = COUNTRY,

CUSTOMERS.CONTENT = CONTENT

**WHERE** CUSTOMERS.CUSTOMERID =CTID;

**END** $$

## spCustomer\_Delete

DELIMITER $$

**CREATE** **PROCEDURE** spCustomer\_Delete (**IN** CTID **nvarchar**(15))

**BEGIN**

**DELETE** **FROM** CUSTOMERS

**WHERE** CUSTOMERID = CTID;

**END** $$

## spCustomer\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spCustomer\_GetByValue (**IN** SEARCHVALUE **nvarchar**(50))

**BEGIN**

**SELECT** CT.\* **FROM** CUSTOMERS CT

**LEFT** **JOIN** CUSTOMER\_ORDERS CO **ON** CT.CUSTOMERID = CO.CO\_CTID

**LEFT** **JOIN** ORDERS O **ON** O.ORDERID = CO.CO\_OID

**WHERE** CT.CID **LIKE** SEARCHVALUE

**OR** CT.CUSTOMERID **LIKE** SEARCHVALUE

**OR** CT.FIRST\_NAME **LIKE** SEARCHVALUE

**OR** CT.LAST\_NAME **LIKE** SEARCHVALUE

**OR** CT.CONTACT\_PHONE **LIKE** SEARCHVALUE

**OR** CT.SL\_ADDRESS **LIKE** SEARCHVALUE

**OR** CO.CO\_ID **LIKE** SEARCHVALUE

**OR** O.ORDERID **LIKE** SEARCHVALUE

**OR** O.CREATED\_AT **LIKE** SEARCHVALUE

**ORDER** **BY** CT.CUSTOMERID;

**END** $$

## spOrder\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spOrder\_Insert (

**IN** CREATED\_BY **NVARCHAR**(9),

**IN** CREATED\_AT **DATETIME**,

**IN** UPDATED\_AT **DATETIME**,

**IN** ORDER\_TYPE **SMALLINT**,

**IN** PAYMENT\_METHOD **SMALLINT**,

**IN** TOTAL **FLOAT**,

**IN** GRAND\_TOTAL **FLOAT**,

**IN** STATUS **SMALLINT**,

**IN** COMMENTS **NVARCHAR**(255),

**OUT** OID **INT**

)

**BEGIN**

-- set default values

**IF** ORDER\_TYPE **IS** **NULL** **THEN** **SET** ORDER\_TYPE = 0; **END** **IF**;

**IF** PAYMENT\_METHOD **IS** **NULL** **THEN** **SET** PAYMENT\_METHOD = 0; **END** **IF**;

**IF** CREATED\_AT **IS** **NULL** **THEN** **SET** CREATED\_AT = **CURRENT\_TIMESTAMP**; **END** **IF**;

**IF** STATUS **IS** **NULL** **THEN** **SET** STATUS = 0; **END** **IF**;

-- Insert statements for procedure here

**INSERT** **INTO** ORDERS(CREATED\_BY,CREATED\_AT,UPDATED\_AT,ORDER\_TYPE,PAYMENT\_METHOD,TOTAL,GRAND\_TOTAL,STATUS,COMMENTS)

**VALUES** (CREATED\_BY,CREATED\_AT,UPDATED\_AT,ORDER\_TYPE,PAYMENT\_METHOD,TOTAL,GRAND\_TOTAL,STATUS,COMMENTS);

**SET** OID = (**SELECT** LAST\_INSERT\_ID());

**END** $$

## spOrder\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spOrder\_Update (

**IN** ORDERID **INT**,

**IN** CREATED\_BY **NVARCHAR**(9),

**IN** CREATED\_AT **DATETIME**,

**IN** UPDATED\_AT **DATETIME**,

**IN** ORDER\_TYPE **SMALLINT**,

**IN** PAYMENT\_METHOD **SMALLINT**,

**IN** TOTAL **FLOAT**,

**IN** GRAND\_TOTAL **FLOAT**,

**IN** STATUS **SMALLINT**,

**IN** COMMENTS **NVARCHAR**(255)

)

**BEGIN**

-- set default values

**IF** ORDER\_TYPE **IS** **NULL** **THEN** **SET** ORDER\_TYPE = 0; **END** **IF**;

**IF** PAYMENT\_METHOD **IS** **NULL** **THEN** **SET** PAYMENT\_METHOD = 0; **END** **IF**;

**IF** STATUS **IS** **NULL** **THEN** **SET** STATUS = 0; **END** **IF**;

-- Insert statements for procedure here

**IF** CREATED\_AT **IS** **NOT** **NULL** **THEN**

**UPDATE** ORDERS **SET** ORDERS.CREATED\_AT = CREATED\_AT **WHERE** ORDERS.ORDERID = ORDERID; **END** **IF**;

**UPDATE** ORDERS

**SET**

ORDERS.CREATED\_BY = CREATED\_BY,

ORDERS.UPDATED\_AT = **CURRENT\_TIMESTAMP**,

ORDERS.ORDER\_TYPE = ORDER\_TYPE,

ORDERS.PAYMENT\_METHOD = PAYMENT\_METHOD,

ORDERS.TOTAL = TOTAL,

ORDERS.GRAND\_TOTAL = GRAND\_TOTAL,

ORDERS.STATUS = STATUS,

ORDERS.COMMENTS = COMMENTS

**WHERE** ORDERS.ORDERID = ORDERID;

**END** $$

## spOrder\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spOrder\_GetAll ()

**BEGIN**

**SELECT** \* **FROM** ORDERS ;

**END** $$

## spOrder\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spOrder\_GetByValue (**IN** **VALUE** **nvarchar**(50))

**BEGIN**

**SELECT** **distinct** \* **FROM** ORDERS

**WHERE** **cast**(ORDERID **as** **char**) **LIKE** **VALUE**

**OR** CREATED\_BY **LIKE** **VALUE**

**OR** COMMENTS **LIKE** **VALUE**

**OR** **cast**(TOTAL **as** **char**) **LIKE** **VALUE**

**OR** **cast**(GRAND\_TOTAL **as** **char**)**LIKE** **VALUE**;

**END** $$

## spProduct\_UpdateProductStock

DELIMITER $$

**CREATE** **PROCEDURE** spProduct\_UpdateProductStock (

**IN** OD\_PID **NVARCHAR**(8),

**IN** OLD\_QUANTITY **INT**,

**IN** NEW\_QUANTITY **INT**

)

**BEGIN**

-- Insert statements for procedure here

**UPDATE** PRODUCTS

**SET** PRODUCTS.UNIT\_ON\_STOCK = UNIT\_ON\_STOCK + OLD\_QUANTITY - NEW\_QUANTITY

**WHERE** PRODUCTS.PRODUCTID = OD\_PID;

**END** $$

## spProduct\_UpdateProductOrder

DELIMITER $$

**CREATE** **PROCEDURE** spProduct\_UpdateProductOrder (

**IN** OD\_PID **NVARCHAR**(8),

**IN** OLD\_QUANTITY **INT**,

**IN** NEW\_QUANTITY **INT**

)

**BEGIN**

-- Insert statements for procedure here

**UPDATE** PRODUCTS

**SET** PRODUCTS.UNIT\_ON\_ORDER = UNIT\_ON\_ORDER - OLD\_QUANTITY + NEW\_QUANTITY

**WHERE** PRODUCTS.PRODUCTID = OD\_PID;

**END** $$

## spOrderDetails\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spOrderDetails\_Insert (

**IN** OD\_OID **INT**,

**IN** OD\_PID **NVARCHAR**(8),

**IN** UNIT\_PRICE **FLOAT**,

**IN** QUANTITY **SMALLINT**,

**IN** DISCOUNT **FLOAT**

)

**BEGIN**

-- initialize variables

**DECLARE** NEW\_QUANTITY **INT** **default** 0;

-- set default values

**IF** UNIT\_PRICE **IS** **NULL** **THEN** **SET** UNIT\_PRICE = 0; **END** **IF**;

**IF** QUANTITY **IS** **NULL** **THEN** **SET** QUANTITY = 1; **END** **IF**;

**IF** DISCOUNT **IS** **NULL** **THEN** **SET** DISCOUNT = 0.0; **END** **IF**;

-- Insert statements for procedure here

**INSERT** **INTO** ORDER\_DETAILS(OD\_OID,OD\_PID,UNIT\_PRICE, QUANTITY, DISCOUNT)

**VALUES** (OD\_OID,OD\_PID,UNIT\_PRICE, QUANTITY, DISCOUNT);

**END** $$

## spOrderDetails\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spOrderDetails\_Update (

**IN** OD\_ID **INT**,

**IN** OD\_OID **INT**,

**IN** OD\_PID **NVARCHAR**(8),

**IN** UNIT\_PRICE **FLOAT**,

**IN** QUANTITY **SMALLINT**,

**IN** DISCOUNT **FLOAT**

)

**BEGIN**

-- set default values

**IF** UNIT\_PRICE **IS** **NULL** **THEN** **SET** UNIT\_PRICE = 0; **END** **IF**;

**IF** QUANTITY **IS** **NULL** **THEN** **SET** QUANTITY = 1; **END** **IF**;

**IF** DISCOUNT **IS** **NULL** **THEN** **SET** DISCOUNT = 0.0; **END** **IF**;

-- Insert statements for procedure here

**UPDATE** ORDER\_DETAILS

**SET** ORDER\_DETAILS.OD\_PID = OD\_PID,

ORDER\_DETAILS.UNIT\_PRICE = UNIT\_PRICE,

ORDER\_DETAILS.QUANTITY = QUANTITY,

ORDER\_DETAILS.DISCOUNT = DISCOUNT

**WHERE** ORDER\_DETAILS.OD\_ID = OD\_ID;

**END** $$

## spOrderDetails\_GetByID

DELIMITER $$

**CREATE** **PROCEDURE** spOrderDetails\_GetByID (**IN** OD\_ID **INT**)

**BEGIN**

-- Insert statements for procedure here

**SELECT** \* **FROM** ORDER\_DETAILS

**WHERE** ORDER\_DETAILS.OD\_ID = OD\_ID;

**END** $$

## spOrderDetail\_GetByID

DELIMITER $$

**CREATE** **PROCEDURE** spOrderDetails\_GetByOrder (**IN** OD\_OID **INT**)

**BEGIN**

-- Insert statements for procedure here

**SELECT** \* **FROM** ORDER\_DETAILS

**WHERE** ORDER\_DETAILS.OD\_OID = OD\_OID;

**END** $$

## spOrderDetails\_Delete

DELIMITER $$

**CREATE** **PROCEDURE** spOrderDetails\_Delete (**IN** OD\_ID **INT**)

**BEGIN**

**DELETE** **FROM** ORDER\_DETAILS

**WHERE** ORDER\_DETAILS.OD\_ID = OD\_ID ;

**END** $$

## spStockOrder\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spStockOrder\_Insert (

**IN** SO\_SLID **NVARCHAR** (8),

**IN** SO\_OID **INT**

)

**BEGIN**

**INSERT** **INTO** SUPPLY\_STOCKS (SO\_SLID, SO\_OID)

**VALUES** (SO\_SLID,SO\_OID);

**END** $$

## spStockOrder\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spStockOrder\_Update (

**in** SO\_ID **INT**,

**in** SO\_SLID **NVARCHAR**(8),

**in** SO\_OID **INT**

)

**BEGIN**

**UPDATE** SUPPLY\_STOCKS

**SET** SUPPLY\_STOCKS.SO\_OID = SO\_OID,

SUPPLY\_STOCKS.SO\_SLID = SO\_SLID

**WHERE** SUPPLY\_STOCKS.SO\_ID = SO\_ID;

**END** $$

## spStockOrder\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spStockOrder\_GetAll ()

**BEGIN**

**SELECT** SO\_ID, SO\_SLID, O.\* **FROM** SUPPLY\_STOCKS SS

**JOIN** ORDERS O **ON** SS.SO\_OID = O.ORDERID;

**END** $$

## spStockOrder\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spStockOrder\_GetByValue (**IN** **VALUE** **NVARCHAR**(50))

**BEGIN**

**SELECT** **DISTINCT** SO\_ID, SO\_SLID, O.\* **FROM** SUPPLY\_STOCKS SS

**JOIN** ORDERS O **ON** SS.SO\_OID = O.ORDERID

**JOIN** SUPPLIERS S **ON** S.SUPPLIERID = SS.SO\_SLID

**WHERE** **cast**(SO\_ID **as** **char**) **LIKE** **VALUE**

**OR** SO\_SLID **LIKE** **VALUE**

**OR** **cast**(ORDERID **as** **char**) **LIKE** **VALUE**

**OR** **cast**(TOTAL **as** **char**) **LIKE** **VALUE**

**OR** **cast**(GRAND\_TOTAL **as** **char**) **LIKE** **VALUE**

**OR** CREATED\_BY **LIKE** **VALUE**

**OR** COMMENTS **LIKE** **VALUE**

**OR** SUPPLIERID **LIKE** **VALUE**

**OR** SUPPLIER\_NAME **LIKE** **VALUE**

**OR** CONTACT\_NAME **LIKE** **VALUE**

**OR** CONTACT\_PHONE **LIKE** **VALUE**

**OR** SL\_ADDRESS **LIKE** **VALUE**

**OR** CITY **LIKE** **VALUE**

**OR** COUNTRY **LIKE** **VALUE**;

**END** $$

## spCustomerOrder\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spCustomerOrder\_Insert (

**IN** CO\_CTID **NVARCHAR**(8),

**IN** CO\_OID **INT**

)

**BEGIN**

**INSERT** **INTO** CUSTOMER\_ORDERS(CO\_CTID,CO\_OID)

**VALUES** (CO\_CTID,CO\_OID);

**END** $$

## spCustomerOrder\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spCustomerOrder\_Update (

**in** CO\_ID **INT**,

**in** CO\_CTID **NVARCHAR**(8),

**in** CO\_OID **INT**

)

**BEGIN**

**UPDATE** CUSTOMER\_ORDERS

**SET** CUSTOMER\_ORDERS.CO\_OID = CO\_OID,

CUSTOMER\_ORDERS.CO\_CTID = CO\_CTID

**WHERE** CUSTOMER\_ORDERS.CO\_ID = CO\_ID;

**END** $$

## spCustomerOrder\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spCustomerOrder\_GetAll ()

**BEGIN**

**SELECT** CO\_ID, CO\_CTID, O.\* **FROM** CUSTOMER\_ORDERS

**JOIN** ORDERS O **ON** CO\_OID = ORDERID;

**END** $$

## spCustomerOrder\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spCustomerOrder\_GetByValue (**IN** **VALUE** **NVARCHAR**(50))

**BEGIN**

**SELECT** **DISTINCT** CO\_ID, CO\_CTID, O.\* **FROM** CUSTOMER\_ORDERS CO

**JOIN** ORDERS O **ON** ORDERID = CO\_OID

**JOIN** CUSTOMERS C **ON** C.CUSTOMERID = CO\_CTID

**WHERE** **cast**(CO\_ID **as** **char**) **LIKE** **VALUE**

**OR** CO\_CTID **LIKE** **VALUE**

**OR** **cast**(ORDERID **as** **char**) **LIKE** **VALUE**

**OR** CREATED\_BY = **VALUE**

**OR** CID = **VALUE**

**OR** **concat**(FIRST\_NAME,' ', LAST\_NAME) **LIKE** **concat**('%',**VALUE**,'%')

**OR** SL\_ADDRESS **LIKE** **VALUE**

**OR** CITY **LIKE** **VALUE**

**OR** COUNTRY **LIKE** **VALUE**

**OR** CONTACT\_PHONE **LIKE** **VALUE**

**OR** COMMENTS = **VALUE**;

**END** $$

## spAccount\_GetByUID

DELIMITER $$

**CREATE** **PROCEDURE** spAccount\_GetByUID (**IN** **UID** **NVARCHAR**(50))

**BEGIN**

**IF** **EXISTS**(**SELECT** 1 **FROM** ACCOUNTS A **WHERE** A.USERID = **UID**) **then**

**BEGIN**

-- TAKE WHOLE MODEL INCLUDING USER ROLES

**SELECT** A.\*,R.\* **FROM** ACCOUNTS A

**LEFT** **JOIN** ACCOUNT\_ROLEVIEWS **ON** AR\_UID = A.USERID

**LEFT** **JOIN** ROLEVIEWS R **ON** R.RID = AR\_RID

**WHERE** USERID = **UID** ;

**END**;

**end** **if**;

**END** $$

## spAccount\_GetByUsername

DELIMITER $$

**CREATE** **PROCEDURE** spAccount\_GetByUsername (**IN** USERNAME **NVARCHAR**(50))

**BEGIN**

**IF** **EXISTS**(**SELECT** 1 **FROM** ACCOUNTS A **WHERE** A.USERNAME = USERNAME ) **then**

**BEGIN**

-- UPDATE THE THE LOGIN TIME BY THE CURRENT TIME

**UPDATE** ACCOUNTS

**SET**

LAST\_LOGIN = NOW()

**WHERE** ACCOUNTS.USERNAME = USERNAME;

-- TAKE WHOLE MODEL INCLUDING USER ROLES

**SELECT** A.\*,R.\* **FROM** ACCOUNTS A

**LEFT** **JOIN** ACCOUNT\_ROLEVIEWS **ON** AR\_UID = A.USERID

**LEFT** **JOIN** ROLEVIEWS R **ON** R.RID = AR\_RID

**WHERE** A.USERNAME = USERNAME ;

**END**;

**end** **if**;

**END** $$

## spAccountRole\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spAccountRole\_GetAll ()

**BEGIN**

**SELECT** \* **FROM** DBO.ACCOUNT\_ROLES;

**END** $$

## spEmployee\_Delete

DELIMITER $$

**CREATE** **PROCEDURE** spEmployee\_Delete (**IN** EMPLOYEEID **nvarchar**(15))

**BEGIN**

**DELETE** **FROM** EMPLOYEES

**WHERE** EMPLOYEES.EMPLOYEEID = EMPLOYEEID;

**END** $$

## spEmployee\_GetAll

DELIMITER $$

**CREATE** **PROCEDURE** spEmployee\_GetAll ()

**BEGIN**

**SELECT** E.\*, EC.EC\_UID **FROM** EMPLOYEES E

**LEFT** **JOIN** EMPLOYEE\_ACCOUNTS EC **ON** E.EMPLOYEEID =EC\_EID;

**END** $$

## spEmployee\_GetByValue

DELIMITER $$

**CREATE** **PROCEDURE** spEmployee\_GetByValue (**IN** SEARCHVALUE **nvarchar**(50))

**BEGIN**

**SELECT** **DISTINCT** E.\*, EC.EC\_UID **FROM** EMPLOYEES E

**LEFT** **JOIN** EMPLOYEE\_ACCOUNTS EC **ON** E.EMPLOYEEID = EC\_EID

**LEFT** **JOIN** ACCOUNT\_ROLEVIEWS AR **ON** AR\_UID = EC.EC\_UID

**WHERE** E.CID **LIKE** SEARCHVALUE

**OR** EMPLOYEEID = SEARCHVALUE

**OR** EC.EC\_UID **LIKE** SEARCHVALUE

**OR** E.FIRST\_NAME **LIKE** SEARCHVALUE

**OR** E.LAST\_NAME **LIKE** SEARCHVALUE

**OR** E.PHONE **LIKE** SEARCHVALUE

**OR** E.ADDRESS **LIKE** SEARCHVALUE

**OR** E.CITY **LIKE** SEARCHVALUE

**OR** E.COUNTRY **LIKE** SEARCHVALUE

**OR** E.CONTENT **LIKE** SEARCHVALUE

**OR** E.TITLE **LIKE** SEARCHVALUE

**ORDER** **BY** E.EMPLOYEEID **DESC**;

**END** $$

## spEmployee\_Insert

DELIMITER $$

**CREATE** **PROCEDURE** spEmployee\_Insert (

**IN** CID **NVARCHAR**(11),

**IN** FIRST\_NAME **NVARCHAR**(15) ,

**IN** LAST\_NAME **NVARCHAR**(15) ,

**IN** BIRTHDATE **DATETIME**,

**IN** SEX **SMALLINT**,

**IN** TITLE **NVARCHAR**(25),

**IN** HIDE\_DATE **DATETIME** ,

**IN** ADDRESS **NVARCHAR**(50) ,

**IN** CITY **NVARCHAR**(20) ,

**IN** COUNTRY **NVARCHAR** (25) ,

**IN** PHONE **VARCHAR**(20) ,

**IN** UPDATE\_AT **DATETIME** ,

**IN** CONTENT **NVARCHAR**(255)

)

**BEGIN**

-- initialize variables

**DECLARE** NEWSLID **nvarchar**(8);

**DECLARE** CURRENTSLID **nvarchar**(8);

**DECLARE** **NEWID** **INT**;

-- set default values

**IF** NEWSLID **IS** **NULL** **THEN** **SET** NEWSLID = 'EL1000'; **END** **IF**;

**IF** SEX **IS** **NULL** **THEN** **SET** SEX = 0; **END** **IF**;

-- CHECK TABLE IS CONTAIN ROWS

**IF** **EXISTS** (**SELECT** 1 **FROM** EMPLOYEES) **THEN**

**BEGIN**

-- TAKE THE LARGEST VALUE OF ID AND PLUS ONE IN FORMAT

**SET** CURRENTSLID = (**Select** **Max**(EMPLOYEEID) **from** EMPLOYEES C);

**SET** **NEWID** = (**SELECT** **Cast**(**SUBSTRING**(CURRENTSLID,3,**LENGTH**(CURRENTSLID)) **AS** UNSIGNED)+1) ;

**SET** NEWSLID = **Concat**('EL',**CAST**(**NEWID** **as** **char**));

**END**;

**END** **IF**;

-- Insert statements for procedure here

**INSERT** **INTO** EMPLOYEES(EMPLOYEEID,CID,FIRST\_NAME, LAST\_NAME,BIRTHDATE,TITLE,HIDE\_DATE, PHONE, ADDRESS, CITY, COUNTRY ,UPDATE\_AT, CONTENT,SEX)

**VALUES** (NEWSLID,CID,FIRST\_NAME, LAST\_NAME,BIRTHDATE,TITLE,HIDE\_DATE, PHONE, ADDRESS ,CITY, COUNTRY ,UPDATE\_AT, CONTENT,SEX);

**END** $$

## spEmployee\_Update

DELIMITER $$

**CREATE** **PROCEDURE** spEmployee\_Update (

**IN** EMPLOYEEID **NVARCHAR**(8), -- EL0001

**IN** CID **NVARCHAR**(11),

**IN** FIRST\_NAME **NVARCHAR**(15) ,

**IN** LAST\_NAME **NVARCHAR**(15) ,

**IN** BIRTHDATE **DATETIME**,

**IN** SEX **SMALLINT**,

**IN** TITLE **NVARCHAR**(25),

**IN** HIDE\_DATE **DATETIME** ,

**IN** ADDRESS **NVARCHAR**(50) ,

**IN** CITY **NVARCHAR**(20) ,

**IN** COUNTRY **NVARCHAR** (25) ,

**IN** PHONE **VARCHAR**(20) ,

**IN** UPDATE\_AT **DATETIME** ,

**IN** CONTENT **NVARCHAR**(255)

)

**BEGIN**

-- set default values

**IF** SEX **IS** **NULL** **THEN** **SET** SEX = 0; **END** **IF**;

-- Insert statements for procedure here

**UPDATE** EMPLOYEES

**SET** EMPLOYEES.CID = CID,

EMPLOYEES.FIRST\_NAME = FIRST\_NAME,

EMPLOYEES.LAST\_NAME = LAST\_NAME,

EMPLOYEES.BIRTHDATE = BIRTHDATE,

EMPLOYEES.SEX = SEX,

EMPLOYEES.TITLE= TITLE,

EMPLOYEES.HIDE\_DATE = HIDE\_DATE,

EMPLOYEES.ADDRESS= ADDRESS,

EMPLOYEES.CITY = CITY,

EMPLOYEES.COUNTRY = COUNTRY,

EMPLOYEES.PHONE = PHONE,

EMPLOYEES.UPDATE\_AT = UPDATE\_AT,

EMPLOYEES.CONTENT = CONTENT

**WHERE** EMPLOYEES.EMPLOYEEID = EMPLOYEEID;

**END** $$