I. Physical database

Table T\_CUSTOMER

|  |  |  |  |
| --- | --- | --- | --- |
| Field\_Name | Datatype | Constraints | Note |
| CUSTOMER\_ID | VARCHAR (10) | PRIMARY KEY | ID KHÁCH HÀNG |
| CUSTOMER\_NAME | NVARCHAR (50) | NOT NULL | TÊN KHÁCH HÀNG |
| CUSTOMER\_ADDRESS | NVARCHAR (50) | NOT NULL | ĐỊA CHỈ |
| CITY | NVARCHAR (50) | NOT NULL | THÀNH PHỐ |
| STATE | NVARCHAR (50) | NOT NULL | BANG |
| POSTAL\_CODE | VARCHAR (10) | NOT NULL | MÃ BƯU ĐIỆN |

Table T\_ORDER

|  |  |  |  |
| --- | --- | --- | --- |
| Field\_Name | Datatype | Constraints | Note |
| ORDER \_ID | VARCHAR (10) | PRIMARY KEY | ID ĐƠN ĐẶT HÀNG |
| ORDER \_DATE | DATE | NOT NULL | NGÀY ĐẶT HÀNG |
| CUSTOMER\_ID | VARCHAR (10) | FOREIGN KEY | ID KHÁCH HÀNG |

Table T\_PRODUCT

|  |  |  |  |
| --- | --- | --- | --- |
| Field\_Name | Datatype | Constraints | Note |
| PRODUCT \_ID | VARCHAR (10) | PRIMARY KEY | ID SẢN PHẨM |
| PRODUCT \_DESCRIPTION | NVARCHAR (50) | NOT NULL | MÔ TẢ |
| PRODUCT\_FINISH | DATE | NOT NULL | NGÀY KẾT THÚC |
| STANDARD\_PRICE | FLOAT | >=0 | GIÁ CHUẨN |
| PRODUCT \_LINE\_ID | VARCHAR (10) | NOT NULL | ID SẢN PHẨM LINE |

Table T\_ORDER\_LINE

|  |  |  |  |
| --- | --- | --- | --- |
| Field\_Name | Datatype | Constraints | Note |
| ORDER\_ID | VARCHAR (10) | FOREIGN KEY | ID ĐẶT HÀNG |
| PRODUCT\_ID | VARCHAR (10) | FOREIGN KEY | ID SẢN PHẨM |
| ORDERED\_QUANTITY | INT | >=0 | SỐ LƯỢNG SẢN PHẨM |

I. CREATE TABLE

CREATE TABLE T\_CUSTOMER (

CUSTOMER\_ID VARCHAR(10),

CUSTOMER\_NAME NVARCHAR(50) NOT NULL,

CUSTOMER\_ADDRESS NVARCHAR(50) NOT NULL,

CITY NVARCHAR(50) NOT NULL,

STATE NVARCHAR(50) NOT NULL,

POSTAL\_CODE VARCHAR(10) NOT NULL,

CONSTRAINT pk\_CUSTOMER\_ID PRIMARY KEY(CUSTOMER\_ID)

);

CREATE TABLE T\_ORDER (

ORDER\_ID VARCHAR(10),

ORDER\_DATE DATE NOT NULL,

CUSTOMER\_ID VARCHAR(10) NOT NULL,

CONSTRAINT pk\_ORDER\_ID PRIMARY KEY(ORDER\_ID),

CONSTRAINT fk\_CUSTOMER\_ID FOREIGN KEY(CUSTOMER\_ID) REFERENCES T\_CUSTOMER(CUSTOMER\_ID)

);

CREATE TABLE T\_PRODUCT (

PRODUCT\_ID VARCHAR(10) ,

PRODUCT\_DESCRIPTION NVARCHAR(100) NOT NULL,

PRODUCT\_FINISH NVARCHAR(10) NOT NULL,

STANDARD\_PRICE FLOAT NOT NULL,

STATE NVARCHAR(10) NOT NULL,

PRODUCT\_LINE\_ID VARCHAR(10) NOT NULL,

CONSTRAINT pk\_PRODUCT\_ID PRIMARY KEY(PRODUCT\_ID),

CONSTRAINT ch\_STANDARD\_PRICE CHECK(STANDARD\_PRICE>=0)

);

CREATE TABLE T\_ORDER\_LINE (

ORDER\_ID VARCHAR(10),

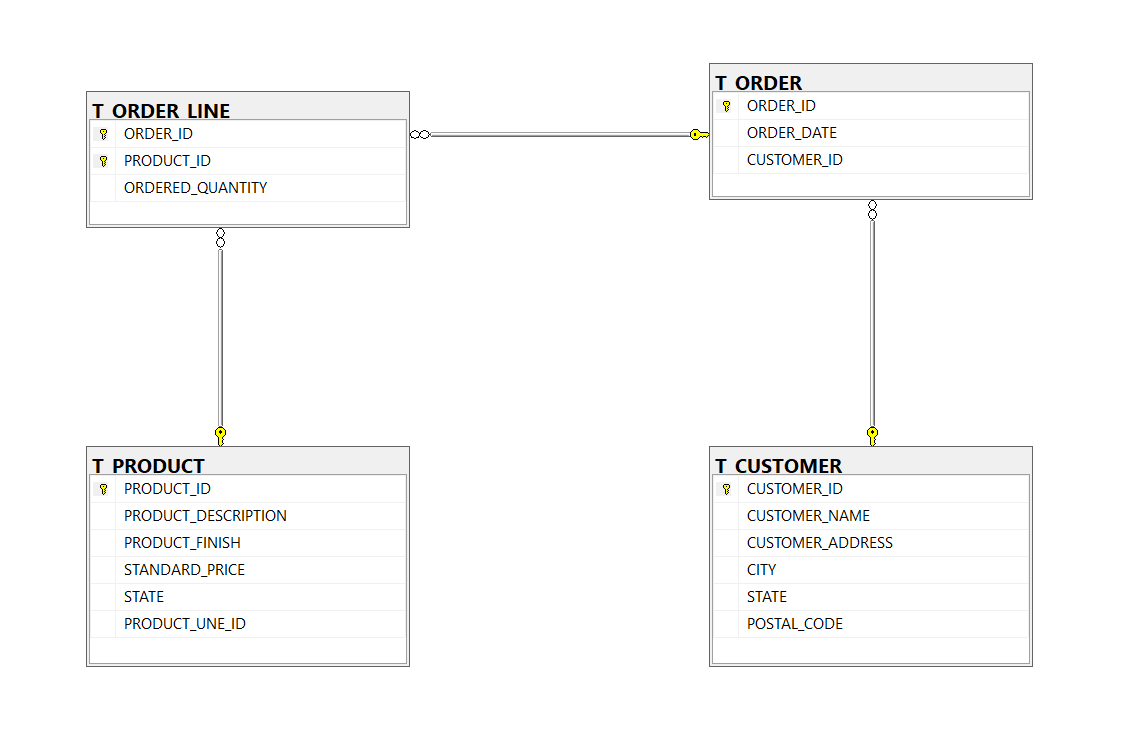
PRODUCT\_ID VARCHAR(10),

ORDERED\_QUANTITY INT NOT NULL,

CONSTRAINT pk\_ORDER\_UNE PRIMARY KEY(ORDER\_ID,PRODUCT\_ID), CONSTRAINT fk\_ORDER\_ID FOREIGN KEY(ORDER\_ID) REFERENCES T\_ORDER(ORDER\_ID),

CONSTRAINT fk\_PRODUCT\_ID FOREIGN KEY(PRODUCT\_ID) REFERENCES T\_PRODUCT(PRODUCT\_ID),

CONSTRAINT ch\_ORDERED\_QUANTITY CHECK(ORDERED\_QUANTITY>=0));



II. INSERT VALUES

--2. Thêm dữ liệu vào bảng bằng INSERT INTO

INSERT INTO T\_CUSTOMER VALUES

(N'C001', N'Nguyễn Văn A', N'Cẩm Lệ', N'Đà Nẵng', N'Bình Thuận', N'99999' ),

(N'C002', N'Nguyễn Văn B', N'Thanh Khê', N'Đà Nẵng', N'Thạc Gián', N'99999' ),

(N'C003', N'Nguyễn Văn C', N'Ngũ Hành Sơn', N'Đà Nẵng', N'Thanh Bình', N'99999' ),

(N'C004', N'Phạm Thị C', N'Hòa Vang', N'Đà Nẵng', N'An Khê', N'99999' ),

(N'C005', N'Nguyễn Thị D', N'Sơn Trà', N'Đà Nẵng', N'Thuận Phước', N'99999' );

INSERT INTO T\_PRODUCT VALUES

(N'P001', N'Gà truyền thống', N'06/10/2024', 35000, N'Còn hàng', N'LINE001' ),

(N'P002', N'Gà sốt cay', N'06/10/2024', 30000, N'Còn hàng', N'LINE002' ),

(N'P003', N'Pepsi', N'06/10/2024', 10000, N'Còn hàng', N'LINE003' ),

(N'P004', N'Kem', N'06/10/2024', 5000, N'Còn hàng', N'LINE004' ),

(N'P005', N'Cơm', N'06/10/2024', 40000, N'Còn hàng', N'LINE005' );

INSERT INTO T\_ORDER VALUES

(N'O001', '2024-06-08', N'C001'),

(N'O002', '2024-06-09', N'C002'),

(N'O003', '2024-06-10', N'C001'),

(N'O004', '2024-06-10', N'C003'),

(N'O005', '2024-06-10', N'C005');

INSERT INTO T\_ORDER\_LINE VALUES

(N'O001', N'P001', 2),

(N'O002', N'P001', 1),

(N'O003', N'P005', 2),

(N'O004', N'P002', 4),

(N'O005', N'P004', 5);

III. QUERY STATEMENTS

--1. Hiển thị các sản phẩm còn hàng

SELECT \* FROM T\_PRODUCT

WHERE STATE = N'Còn hàng';

--2. Hiển thị đơn hàng của khách hàng có mã C001

SELECT \* FROM T\_ORDER WHERE CUSTOMER\_ID = N'C001';

--3. Hiển thị tổng số lượng sản phẩm đã được đặt hàng trong bảng T\_ORDER\_LINE

SELECT PRODUCT\_ID, SUM(ORDERED\_QUANTITY) AS TOTALQUANTITY

FROM T\_ORDER\_LINE

GROUP BY PRODUCT\_ID;

--4. Hiển thị các khách hàng thuộc STATE = ‘Bình Thuận’

SELECT \* FROM T\_CUSTOMER WHERE STATE = N'Bình Thuận';

--5. Hiển thị các đơn hàng được đặt vào ngày '2024-06-10'

SELECT \* FROM T\_ORDER WHERE ORDER\_DATE = '2024-06-10';

--6. Hiển thị tổng giá trị các đơn hàng của từng khách hàng

SELECT C.CUSTOMER\_ID, C.CUSTOMER\_NAME, SUM(P.STANDARD\_PRICE \* OL.ORDERED\_QUANTITY) AS TOTAL\_ORDER\_VALUE

FROM T\_CUSTOMER C

JOIN T\_ORDER O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

JOIN T\_ORDER\_LINE OL ON O.ORDER\_ID = OL.ORDER\_ID

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

GROUP BY C.CUSTOMER\_ID, C.CUSTOMER\_NAME;

--7. Hiển thị danh sách các khách hàng và các đơn hàng của họ

SELECT C.CUSTOMER\_ID, C.CUSTOMER\_NAME, O.ORDER\_ID, O.ORDER\_DATE

FROM T\_CUSTOMER C

JOIN T\_ORDER O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID;

--8. Hiển thị các sản phẩm đã được đặt nhiều nhất

SELECT P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION, SUM(OL.ORDERED\_QUANTITY) AS TOTAL\_ORDER

FROM T\_PRODUCT P

JOIN T\_ORDER\_LINE OL ON P.PRODUCT\_ID = OL.PRODUCT\_ID

GROUP BY P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION

ORDER BY TOTAL\_ORDER DESC;

--9. Hiển thị chi tiết đơn hàng của khách hàng

SELECT O.CUSTOMER\_ID, C.CUSTOMER\_NAME, P.PRODUCT\_DESCRIPTION, OL.ORDERED\_QUANTITY, (P.STANDARD\_PRICE \* OL.ORDERED\_QUANTITY) AS TOTAL\_PRICE

FROM T\_ORDER O JOIN T\_CUSTOMER C

ON O.CUSTOMER\_ID = C.CUSTOMER\_ID

JOIN T\_ORDER\_LINE OL

ON O.ORDER\_ID = OL.ORDER\_ID

JOIN T\_PRODUCT P

ON P.PRODUCT\_ID = OL.PRODUCT\_ID

WHERE O.CUSTOMER\_ID = N'C001';

--10. Hiển thị tổng số lượng đặt hàng và tổng giá trị của mỗi khách hàng

SELECT C.CUSTOMER\_ID,C.CUSTOMER\_NAME,

COUNT(O.ORDER\_ID) AS TOTAL\_ORDERS,

SUM(P.STANDARD\_PRICE \* OL.ORDERED\_QUANTITY) AS TOTAL\_ORDER\_VALUE

FROM T\_CUSTOMER C

JOIN T\_ORDER O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

JOIN T\_ORDER\_LINE OL ON O.ORDER\_ID = OL.ORDER\_ID

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

GROUP BY C.CUSTOMER\_ID, C.CUSTOMER\_NAME

ORDER BY TOTAL\_ORDER\_VALUE DESC;

--11. Hiển thị các sản phẩm được đặt hàng nhiều nhất và số lượng khách hàng đã đặt sản phẩm đó

SELECT P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION,

SUM(OL.ORDERED\_QUANTITY) AS TOTAL\_ORDERED\_QUANTITY,

COUNT(DISTINCT O.CUSTOMER\_ID) AS CUSTOMER\_COUNT

FROM T\_PRODUCT P

JOIN T\_ORDER\_LINE OL ON P.PRODUCT\_ID = OL.PRODUCT\_ID

JOIN T\_ORDER O ON OL.ORDER\_ID = O.ORDER\_ID

GROUP BY P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION

ORDER BY TOTAL\_ORDERED\_QUANTITY DESC;

--12. Hiển thị các khách hàng có tổng giá trị đơn hàng lớn nhất

SELECT C.CUSTOMER\_ID,C.CUSTOMER\_NAME,

SUM(P.STANDARD\_PRICE \* OL.ORDERED\_QUANTITY) AS TOTAL\_ORDER\_VALUE

FROM T\_CUSTOMER C

JOIN T\_ORDER O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

JOIN T\_ORDER\_LINE OL ON O.ORDER\_ID = OL.ORDER\_ID

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

GROUP BY C.CUSTOMER\_ID, C.CUSTOMER\_NAME

ORDER BY TOTAL\_ORDER\_VALUE DESC

--13. Hiển thị số lượng đơn hàng của mỗi khách hàng

SELECT ORDER\_DATE, SUM(ORDERED\_QUANTITY) AS TOTAL\_QUANTITY

FROM T\_ORDER\_LINE OL

JOIN T\_ORDER O ON OL.ORDER\_ID = O.ORDER\_ID

GROUP BY ORDER\_DATE;

--14. Hiển thị các khách hàng đã đặt ít nhất một đơn hàng

SELECT C.CUSTOMER\_ID, C.CUSTOMER\_NAME

FROM T\_CUSTOMER C

WHERE EXISTS (SELECT 1 FROM T\_ORDER O WHERE O.CUSTOMER\_ID = C.CUSTOMER\_ID);

--15. Hiển thị các khách hàng có hơn 2 đơn hàng

SELECT CUSTOMER\_ID, COUNT(ORDER\_ID) AS TOTAL\_ORDERS

FROM T\_ORDER

GROUP BY CUSTOMER\_ID

HAVING COUNT(ORDER\_ID) > 2;

--16. Hiển thị số lượng sản phẩm đã đặt theo từng ngày

SELECT ORDER\_DATE, SUM(ORDERED\_QUANTITY) AS TOTAL\_QUANTITY

FROM T\_ORDER\_LINE OL

JOIN T\_ORDER O ON OL.ORDER\_ID = O.ORDER\_ID

GROUP BY ORDER\_DATE;

--17. Hiển thị các sản phẩm có giá trị đơn hàng tổng cộng lớn hơn 100,000

SELECT P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION, SUM(OL.ORDERED\_QUANTITY \* P.STANDARD\_PRICE) AS TOTAL\_VALUE

FROM T\_PRODUCT P

JOIN T\_ORDER\_LINE OL ON P.PRODUCT\_ID = OL.PRODUCT\_ID

GROUP BY P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION

HAVING SUM(OL.ORDERED\_QUANTITY \* P.STANDARD\_PRICE) > 100000;

--18. Hiển thị các khách hàng đã đặt hàng và tổng số lượng sản phẩm đã đặt của họ

SELECT C.CUSTOMER\_ID, C.CUSTOMER\_NAME, SUM(OL.ORDERED\_QUANTITY) AS TOTAL\_ORDERED\_QUANTITY

FROM T\_CUSTOMER C

JOIN T\_ORDER O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

JOIN T\_ORDER\_LINE OL ON O.ORDER\_ID = OL.ORDER\_ID

GROUP BY C.CUSTOMER\_ID, C.CUSTOMER\_NAME;

--19. Hiển thị các sản phẩm có số lượng đặt hàng trung bình lớn hơn 2

SELECT P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION, AVG(OL.ORDERED\_QUANTITY) AS AVG\_QUANTITY

FROM T\_PRODUCT P

JOIN T\_ORDER\_LINE OL ON P.PRODUCT\_ID = OL.PRODUCT\_ID

GROUP BY P.PRODUCT\_ID, P.PRODUCT\_DESCRIPTION

HAVING AVG(OL.ORDERED\_QUANTITY) > 2;

--20. Hiển thị các khách hàng đã đặt hàng trong tháng 6 năm 2024

SELECT DISTINCT C.CUSTOMER\_ID, C.CUSTOMER\_NAME

FROM T\_CUSTOMER C

JOIN T\_ORDER O ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

WHERE O.ORDER\_DATE BETWEEN '2024-06-01' AND '2024-06-30';

IV. UPDATE & DELETE

--1. Cập nhật tên khách hàng có mã 'C001'

UPDATE T\_CUSTOMER

SET CUSTOMER\_NAME = N'Nguyễn Văn An'

WHERE CUSTOMER\_ID = N'C001';

--2. Cập nhật giá sản phẩm 'P001'

UPDATE T\_PRODUCT

SET STANDARD\_PRICE = 36000

WHERE PRODUCT\_ID = N'P001';

--3. Cập nhật số lượng sản phẩm đặt hàng trong đơn 'O001' và sản phẩm 'P001'

UPDATE T\_ORDER\_LINE

SET ORDERED\_QUANTITY = 3

WHERE ORDER\_ID = N'O001' AND PRODUCT\_ID = N'P001';

--4. Xóa tất cả các đơn hàng có ngày đặt trước '2024-06-10'

DELETE FROM T\_ORDER

WHERE ORDER\_DATE < '2024-06-10';

--5. Xóa tất cả các sản phẩm có giá dưới 10,000

DELETE FROM T\_PRODUCT

WHERE STANDARD\_PRICE < 10000;

--6. Xóa tất cả các đơn hàng không có trong bảng T\_ORDER\_LINE

DELETE FROM T\_ORDER

WHERE ORDER\_ID NOT IN (SELECT ORDER\_ID FROM T\_ORDER\_LINE);

--7. Xóa tất cả các khách hàng không có đơn hàng nào

DELETE FROM T\_CUSTOMER

WHERE CUSTOMER\_ID NOT IN (SELECT CUSTOMER\_ID FROM T\_ORDER);

--8. Cập nhật giá của tất cả các sản phẩm lên 10%

UPDATE T\_PRODUCT

SET STANDARD\_PRICE = STANDARD\_PRICE \* 1.10;

--9. Cập nhật trạng thái sản phẩm 'P003' thành 'Hết hàng'

UPDATE T\_PRODUCT

SET STATE = N'Hết hàng'

WHERE PRODUCT\_ID = N'P003';

--10. Cập nhật giá của tất cả các sản phẩm có giá lớn hơn 30,000 giảm 5%

UPDATE T\_PRODUCT

SET STANDARD\_PRICE = STANDARD\_PRICE \* 0.95

WHERE STANDARD\_PRICE > 30000;

--11. Xóa tất cả các đơn hàng có tổng giá trị nhỏ hơn 50,000

DELETE FROM T\_ORDER

WHERE ORDER\_ID IN (

SELECT O.ORDER\_ID

FROM T\_ORDER O

JOIN T\_ORDER\_LINE OL ON O.ORDER\_ID = OL.ORDER\_ID

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

GROUP BY O.ORDER\_ID

HAVING SUM(P.STANDARD\_PRICE \* OL.ORDERED\_QUANTITY) < 50000

);

--12. Cập nhật trạng thái của tất cả các sản phẩm có số lượng đặt hàng lớn hơn 10 thành 'Hết hàng'

UPDATE T\_PRODUCT

SET STATE = N'Hết hàng'

WHERE PRODUCT\_ID IN (

SELECT PRODUCT\_ID

FROM T\_ORDER\_LINE

GROUP BY PRODUCT\_ID

HAVING SUM(ORDERED\_QUANTITY) > 10

);

--13. Xóa các sản phẩm liên qua đến gà

DELETE FROM T\_PRODUCT

WHERE PRODUCT\_DESCRIPTION LIKE N'%Gà%';

--14. Cập nhật giá của tất cả các sản phẩm dựa trên số lượng đặt hàng của chúng.

-- >5 giá tăng 15%, ngược lại giảm 10%

UPDATE T\_PRODUCT

SET STANDARD\_PRICE = CASE

WHEN (SELECT SUM(ORDERED\_QUANTITY)

FROM T\_ORDER\_LINE

WHERE T\_ORDER\_LINE.PRODUCT\_ID = T\_PRODUCT.PRODUCT\_ID) > 5

THEN STANDARD\_PRICE \* 1.15

ELSE STANDARD\_PRICE \* 0.90

END;

--15. Xóa tất cả các sản phẩm mà chưa từng được đặt hàng

DELETE FROM T\_PRODUCT

WHERE PRODUCT\_ID NOT IN (

SELECT DISTINCT PRODUCT\_ID

FROM T\_ORDER\_LINE

)

--16. Xóa tất cả các khách hàng và các đơn hàng của họ nếu họ không đặt hàng nào trong năm 2024

DELETE FROM T\_ORDER

WHERE CUSTOMER\_ID IN (

SELECT CUSTOMER\_ID

FROM T\_CUSTOMER

WHERE CUSTOMER\_ID NOT IN (

SELECT DISTINCT CUSTOMER\_ID

FROM T\_ORDER

WHERE YEAR(ORDER\_DATE) = 2024

)

);

DELETE FROM T\_CUSTOMER

WHERE CUSTOMER\_ID NOT IN (

SELECT DISTINCT CUSTOMER\_ID

FROM T\_ORDER

WHERE YEAR(ORDER\_DATE) = 2024

);

V. PROCEDURE

--5.1. Viết thủ tục Insert dữ liệu cho các bảng có kiểm tra ràng buộc

--Thủ tục nhập cho bảng T\_PRODUCT

CREATE PROCEDURE SP\_NHAP\_T\_PRODUCT

@PRODUCT\_ID VARCHAR(10), @PRODUCT\_DESCRIPTION NVARCHAR(100), @PRODUCT\_FINISH NVARCHAR(10),

@STANDARD\_PRICE FLOAT, @STATE NVARCHAR(10), @PRODUCT\_UNE\_ID NVARCHAR(10)

AS

BEGIN

IF EXISTS (SELECT \* FROM T\_PRODUCT WHERE PRODUCT\_ID = @PRODUCT\_ID)

BEGIN

PRINT N'TRÙNG MÃ SẢN PHẨM. KIỂM TRA LẠI.'

END

ELSE

IF @STANDARD\_PRICE < 0

PRINT N'GIÁ SẢN PHẨM PHẢI LỚN HƠN 0.'

ELSE

BEGIN

INSERT INTO T\_PRODUCT VALUES (@PRODUCT\_ID, @PRODUCT\_DESCRIPTION, @PRODUCT\_FINISH, @STANDARD\_PRICE, @STATE, @PRODUCT\_UNE\_ID)

PRINT N'THÊM SẢN PHẨM THÀNH CÔNG.'

END

END

SELECT \* FROM T\_PRODUCT;

EXEC SP\_NHAP\_T\_PRODUCT 'P006', N'Gà sốt trứng muối', '06/16/2024', 40000, N'Còn hàng', 'LINE006';

--Thủ tục nhập bảng T\_CUSTOMER

CREATE PROCEDURE SP\_NHAP\_T\_CUSTOMER

@CUSTOMER\_ID VARCHAR(10), @CUSTOMER\_NAME NVARCHAR(50), @CUSTOMER\_ADDRESS NVARCHAR(50),

@CITY NVARCHAR(50), @STATE NVARCHAR(10), @POSTAL\_CODE NVARCHAR(10)

AS

BEGIN

IF EXISTS (SELECT \* FROM T\_CUSTOMER WHERE CUSTOMER\_ID = @CUSTOMER\_ID)

PRINT N'TRÙNG MÃ KHÁCH HÀNG. KIỂM TRA LẠI.'

ELSE

BEGIN

INSERT INTO T\_CUSTOMER VALUES (@CUSTOMER\_ID, @CUSTOMER\_NAME, @CUSTOMER\_ADDRESS, @CITY, @STATE, @POSTAL\_CODE)

PRINT N'THÊM KHÁCH HÀNG THÀNH CÔNG.'

END

END

SELECT \* FROM T\_CUSTOMER;

EXEC SP\_NHAP\_T\_CUSTOMER 'C006', N'Trần Thị E', N'Hải Châu', N'Đà Nẵng', 'Nhơn Hòa', '99999';

--Thủ tục nhập bảng T\_ORDER

CREATE PROCEDURE SP\_NHAP\_T\_ORDER

@ORDER\_ID VARCHAR(10), @ORDER\_DATE DATE, @CUSTOMER\_ID VARCHAR(10)

AS

BEGIN

IF NOT EXISTS (SELECT \* FROM T\_CUSTOMER WHERE CUSTOMER\_ID = @CUSTOMER\_ID)

PRINT N'KHÔNG TỒN TẠI KHÁCH HÀNG NÀY. KIỂM TRA LẠI.'

ELSE

IF EXISTS (SELECT \* FROM T\_ORDER WHERE ORDER\_ID = @ORDER\_ID)

PRINT N'TRÙNG MÃ ĐƠN HÀNG. KIỂM TRA LẠI.'

ELSE

BEGIN

INSERT INTO T\_ORDER VALUES (@ORDER\_ID, @ORDER\_DATE, @CUSTOMER\_ID)

PRINT N'NHẬP ĐƠN ĐẶT HÀNG THÀNH CÔNG.'

END

END

SELECT \* FROM T\_ORDER;

SELECT \* FROM T\_CUSTOMER;

EXEC SP\_NHAP\_T\_ORDER 'O006', '2024-06-12', 'C006';

--Thủ tục nhập bảng T\_ORDER\_LINE

CREATE PROCEDURE SP\_NHAP\_T\_ORDER\_LINE

@ORDER\_ID VARCHAR(10), @PRODUCT\_ID VARCHAR(10), @ORDERED\_QUANTITY INT

AS

BEGIN

IF NOT EXISTS (SELECT \* FROM T\_ORDER WHERE ORDER\_ID = @ORDER\_ID)

PRINT N'KHÔNG TỒN TẠI ĐƠN HÀNG NÀY. KIỂM TRA LẠI.'

ELSE

IF NOT EXISTS (SELECT \* FROM T\_PRODUCT WHERE PRODUCT\_ID = @PRODUCT\_ID)

PRINT N'KHÔNG TỒN TẠI SẢN PHẨM NÀY. KIỂM TRA LẠI.'

ELSE

IF EXISTS (SELECT \* FROM T\_ORDER\_LINE WHERE ORDER\_ID = @ORDER\_ID AND PRODUCT\_ID = @PRODUCT\_ID)

PRINT N'TRÙNG ĐƠN HÀNG. KIỂM TRA LẠI.'

ELSE

IF @ORDERED\_QUANTITY < 0

PRINT N'SỐ LƯỢNG ĐẶT HÀNG PHẢI LỚN HƠN 0'

ELSE

BEGIN

INSERT INTO T\_ORDER\_LINE VALUES (@ORDER\_ID, @PRODUCT\_ID, @ORDERED\_QUANTITY)

PRINT N'THÊM ĐƠN HÀNG THÀNH CÔNG.'

END

END

SELECT \* FROM T\_ORDER\_LINE;

SELECT \* FROM T\_PRODUCT;

EXEC SP\_NHAP\_T\_ORDER\_LINE 'O006', 'P006', 4;

--5.2. Cho biết thông tin hóa đơn của khách hàng khi nhân viên nhập vào MAKH, Ngày mua hàng

CREATE PROCEDURE SP\_THONGTINKH

@CUSTOMER\_ID VARCHAR(10), @ORDER\_DATE DATE

AS

BEGIN

IF NOT EXISTS (SELECT \* FROM T\_CUSTOMER WHERE CUSTOMER\_ID = @CUSTOMER\_ID)

PRINT N'KHÔNG TỒN TẠI KHÁCH HÀNG NÀY.'

ELSE

IF NOT EXISTS (SELECT \* FROM T\_ORDER WHERE CUSTOMER\_ID = @CUSTOMER\_ID)

PRINT N'KHÁCH HÀNG NÀY CHƯA CÓ ĐƠN HÀNG NÀO.'

ELSE

IF NOT EXISTS (SELECT \* FROM T\_ORDER WHERE ORDER\_DATE = @ORDER\_DATE)

PRINT N'KHÔNG CÓ ĐƠN HÀNG NÀO VÀO NGÀY NÀY.'

ELSE

SELECT \* FROM T\_CUSTOMER C

JOIN T\_ORDER O

ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

WHERE O.CUSTOMER\_ID = @CUSTOMER\_ID AND O.ORDER\_DATE = @ORDER\_DATE

END

EXEC SP\_THONGTINKH 'C006', '2024-06-12';

--5.3. Viết thủ tục khi nhập ORDER\_ID hiển thị thông tin khách hàng mua những mặt hàng gì, thành tiền bao nhiêu

CREATE PROCEDURE SP\_THONGTINMUAHANG

@ORDER\_ID VARCHAR(10)

AS

BEGIN

IF NOT EXISTS (SELECT \* FROM T\_ORDER WHERE ORDER\_ID = @ORDER\_ID)

PRINT N'KHÔNG TỒN TẠI ĐƠN HÀNG NÀY. KIỂM TRA LẠI.'

ELSE

SELECT O.ORDER\_ID, C.CUSTOMER\_NAME, P.PRODUCT\_DESCRIPTION,

P.STANDARD\_PRICE, OL.ORDERED\_QUANTITY, (P.STANDARD\_PRICE \* OL.ORDERED\_QUANTITY) AS N'TOTAL\_PRICE' FROM T\_ORDER\_LINE OL

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

JOIN T\_ORDER O ON OL.ORDER\_ID = O.ORDER\_ID

JOIN T\_CUSTOMER C ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

WHERE O.ORDER\_ID = @ORDER\_ID

END

SELECT \* FROM T\_ORDER;

SELECT \* FROM T\_PRODUCT;

SELECT \* FROM T\_ORDER\_LINE;

EXEC SP\_THONGTINMUAHANG 'O001';

--5.4. Viết thủ tục cho biết thông tin những mặt hàng nào được bán ra trong ngày, khi nhập ngày

CREATE PROCEDURE SP\_THONG\_TIN\_HANG\_TRONG\_NGAY

@ORDER\_DATE DATE

AS

BEGIN

IF NOT EXISTS (SELECT \* FROM T\_ORDER WHERE ORDER\_DATE = @ORDER\_DATE)

PRINT N'KHÔNG CÓ ĐƠN HÀNG NÀO ĐƯỢC BÁN RA TRONG NGÀY NÀY.'

ELSE

SELECT P.PRODUCT\_DESCRIPTION, O.ORDER\_DATE

FROM T\_ORDER\_LINE OL

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

JOIN T\_ORDER O ON OL.ORDER\_ID = O.ORDER\_ID

WHERE O.ORDER\_DATE = @ORDER\_DATE

END

EXEC SP\_THONG\_TIN\_HANG\_TRONG\_NGAY '2024-06-10';

--5.5. Viết thủ tục tìm khách hàng mua mặt hàng 'xxxx' trong tháng/năm

CREATE PROCEDURE SP\_THONG\_TIN\_HANG\_TRONG\_THANG

@PRODUCT\_DESCRIPTION NVARCHAR(100), @ORDER\_DATE DATE

AS

BEGIN

IF NOT EXISTS (SELECT \* FROM T\_PRODUCT WHERE PRODUCT\_DESCRIPTION = @PRODUCT\_DESCRIPTION)

PRINT N'KHÔNG CÓ SẢN PHẨM TÊN NÀY.'

ELSE

IF YEAR(@ORDER\_DATE) > YEAR(GETDATE())

PRINT N'NĂM KHÔNG HỢP LỆ. KIỂM TRA LẠI'

ELSE

IF NOT EXISTS (SELECT \* FROM T\_ORDER WHERE MONTH(ORDER\_DATE) = MONTH(@ORDER\_DATE))

PRINT N'KHÔNG CÓ ĐƠN HÀNG NÀO ĐƯỢC BÁN RA TRONG THÁNG NÀY.'

ELSE

SELECT C.CUSTOMER\_NAME, P.PRODUCT\_DESCRIPTION, O.ORDER\_DATE

FROM T\_ORDER\_LINE OL

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

JOIN T\_ORDER O ON OL.ORDER\_ID = O.ORDER\_ID

JOIN T\_CUSTOMER C ON C.CUSTOMER\_ID = O.CUSTOMER\_ID

WHERE MONTH(O.ORDER\_DATE) = MONTH(@ORDER\_DATE) AND YEAR(O.ORDER\_DATE) = YEAR(@ORDER\_DATE) AND P.PRODUCT\_DESCRIPTION = @PRODUCT\_DESCRIPTION

END

EXEC SP\_THONG\_TIN\_HANG\_TRONG\_THANG N'Cơm', '2024-06-30';

VI. TRIGGER

--6.1. Tạo bảng

CREATE TABLE INVOICE(

INVOICE\_NUMBER INT,

ORDER\_ID VARCHAR(10) NOT NULL,

INVOICE\_DATE DATE NOT NULL,

CONSTRAINT pk\_INVOICE\_NUMBER PRIMARY KEY(INVOICE\_NUMBER),

CONSTRAINT fk\_ORDER\_ID\_INVOICE FOREIGN KEY(ORDER\_ID) REFERENCES T\_ORDER(ORDER\_ID),

CONSTRAINT uq\_ORDER\_ID UNIQUE(ORDER\_ID)

);

create table PAYMENT(

INVOICE\_NUMBER INT NOT NULL,

PAYMENT\_DATE DATE,

PAYMENT\_AMOUNT FLOAT NOT NULL,

CONSTRAINT pk\_PAYMENT\_DATE PRIMARY KEY(PAYMENT\_DATE),

CONSTRAINT fk\_INVOICE\_NUMBER FOREIGN KEY(INVOICE\_NUMBER) REFERENCES INVOICE(INVOICE\_NUMBER),

);

create table HISTORY\_INVOICE(

INVOICE\_NUMBER INT,

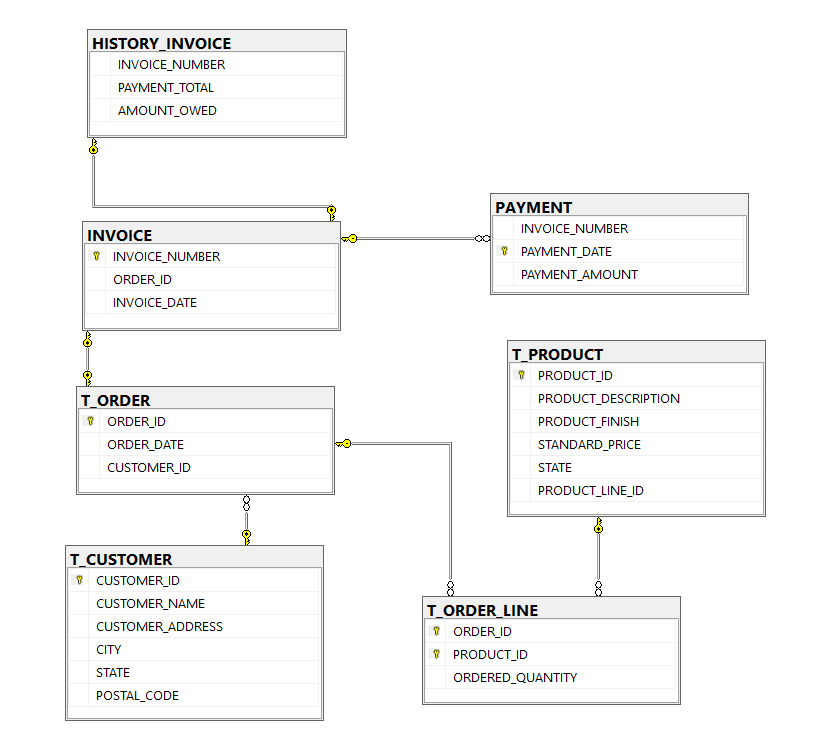
PAYMENT\_TOTAL FLOAT NOT NULL,

AMOUNT\_OWED FLOAT NOT NULL,

CONSTRAINT fk\_INVOICE\_NUMBER\_HISTORY FOREIGN KEY(INVOICE\_NUMBER) REFERENCES INVOICE(INVOICE\_NUMBER),

CONSTRAINT uq\_INVOICE\_NUMBER UNIQUE(INVOICE\_NUMBER)

);



--Viết trigger khi nhập dữ liệu cho bảng invoice thì dữ liệu bảng history\_invoice cũng được thêm vào với pay\_total = amount\_owed = sum (ordered\_quantity\*standard\_price)

CREATE TRIGGER TR\_THEM\_INVOICE

ON INVOICE

FOR INSERT

AS

BEGIN

DECLARE @INVOICE\_NUMBER INT

DECLARE @ORDER\_ID VARCHAR(10)

SELECT @INVOICE\_NUMBER = INVOICE\_NUMBER, @ORDER\_ID = ORDER\_ID FROM INSERTED

IF NOT EXISTS(SELECT \* FROM T\_ORDER WHERE ORDER\_ID = @ORDER\_ID)

BEGIN

RAISERROR('ĐƠN HÀNG NÀY KHÔNG TỒN TẠI. KIỂM TRA LẠI', 16, 1)

ROLLBACK TRANSACTION

RETURN

END

IF EXISTS(SELECT \* FROM INVOICE WHERE INVOICE\_NUMBER = @INVOICE\_NUMBER)

BEGIN

RAISERROR('TRÙNG MÃ HÓA ĐƠN. KIỂM TRA LẠI', 16, 1)

ROLLBACK TRANSACTION

RETURN

END

INSERT INTO HISTORY\_INVOICE (INVOICE\_NUMBER, PAYMENT\_TOTAL, AMOUNT\_OWED)

SELECT I.INVOICE\_NUMBER,

SUM(OL.ORDERED\_QUANTITY \* P.STANDARD\_PRICE) AS PAYMENT\_TOTAL,

SUM(OL.ORDERED\_QUANTITY \* P.STANDARD\_PRICE) AS AMOUNT\_OWED

FROM INSERTED I

JOIN T\_ORDER O ON I.ORDER\_ID = O.ORDER\_ID

JOIN T\_ORDER\_LINE OL ON O.ORDER\_ID = OL.ORDER\_ID

JOIN T\_PRODUCT P ON OL.PRODUCT\_ID = P.PRODUCT\_ID

GROUP BY I.INVOICE\_NUMBER;

END;

SELECT \* FROM T\_ORDER;

SELECT \* FROM INVOICE;

SELECT \* FROM HISTORY\_INVOICE;

INSERT INTO INVOICE VALUES (001, 'O001', '2024-06-08');

--Viết trigger khi khách hàng thanh toán tiền payment\_amount thì số tiền trong amount\_owed cũng giảm theo

CREATE TRIGGER TR\_UPDATE\_AMOUNT\_OWED

ON PAYMENT

FOR INSERT

AS

BEGIN

DECLARE @INVOICE\_NUMBER INT

SET @INVOICE\_NUMBER = (SELECT INVOICE\_NUMBER FROM INSERTED)

IF NOT EXISTS (SELECT \* FROM INVOICE WHERE INVOICE\_NUMBER = @INVOICE\_NUMBER)

BEGIN

RAISERROR('KHÔNG TỒN TẠI HÓA ĐƠN NÀY. KIỂM TRA LẠI', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

UPDATE HISTORY\_INVOICE

SET AMOUNT\_OWED = AMOUNT\_OWED - i.PAYMENT\_AMOUNT

FROM INSERTED i

WHERE HISTORY\_INVOICE.INVOICE\_NUMBER = i.INVOICE\_NUMBER;

END;

SELECT \* FROM HISTORY\_INVOICE;

SELECT \* FROM PAYMENT;

INSERT INTO PAYMENT VALUES (001, '2024-06-08', 35000);

VII. FUNCTION

--Hàm Tính tổng số lượng khách hàng đặt hàng trong ngày,hiển thị những ngày có tổng số lượng khách hàng đặt lớn hơn ngày hiện tại--

CREATE FUNCTION F\_SL\_KHACH\_MUA(@ORDER\_DAY DATE)

RETURNS @KETQUA TABLE (ORDER\_DATE DATE, TOTAL\_CUSTOMER INT)

AS

BEGIN

DECLARE @SOLUONG int

SELECT @SOLUONG = COUNT(DISTINCT CUSTOMER\_ID)

FROM T\_ORDER

WHERE ORDER\_DATE = @ORDER\_DAY

INSERT INTO @KETQUA (ORDER\_DATE, TOTAL\_CUSTOMER)

SELECT ORDER\_DATE, COUNT(DISTINCT CUSTOMER\_ID) AS TOTAL\_CUSTOMER

FROM T\_ORDER

GROUP BY ORDER\_DATE

HAVING COUNT(DISTINCT CUSTOMER\_ID) > @SOLUONG

RETURN

END

SELECT \* FROM DBO.F\_SL\_KHACH\_MUA('2024-06-12')

--Viết hàm để xem tổng số đầu một hàng cho các nhóm hàng khí NSD nhập vào ProductID Biết rằng format của productID là x00000, trong đó X là mã nhóm hàng Xem nhóm hàng điện từ

CREATE FUNCTION F\_DEM\_SL\_HANG(@PRODUCT\_ID VARCHAR(10))

RETURNS INT

AS

BEGIN

DECLARE @SOLUONG INT;

SELECT @SOLUONG = COUNT(DISTINCT PRODUCT\_ID)

FROM T\_PRODUCT

WHERE LEFT(PRODUCT\_LINE\_ID, 1) = LEFT(@PRODUCT\_ID, 1);

RETURN @SOLUONG;

END

SELECT \* FROM T\_PRODUCT;

SELECT DBO.F\_DEM\_SL\_HANG('E00001')