IN PARTNERSHIP WITH PLYMOUTH UNIVERSITY

Name: Kumarapeli Kavishan
Student Reference Number: 10819548

Module Code:	Module Name: Information Management & Retrieval
Coursework Title:	
Deadline Date:	Member of staff responsible for coursework: Mr. Nasiketha Saravanapavan
Programme: BSc (Hons) Software Engineering	

Please note that University Academic Regulations are available under Rules and Regulations on the University website www.plymouth.ac.uk/studenthandbook.

Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.

1. Kumarapeli Kavishan 10819548
2. Deelaka Galpaya 10823134
3. Senarathne Senarathne 10819510
4. Santhushika Rathnayake 10820912
5. Pasan Bopegamage 10818168
6. Vithanage Bandara 10818157

We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.

Signed on behalf of the group:

Individual assignment: ***I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.***

Signed : dulanjana

Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.

I *have used/not used translation software.

If used, please state name of software………………………………………………………………

Overall mark _____% **Assessors Initials _______** **Date__________**

*Please delete as appropriateSci/ps/d:/students/cwkfrontcover/2013/14

Contents

Section 01

Introduction

This project that deals with supermarket automation and it includes both purchasing a selling of items. This project is designed with a goal to making the existing system more informative, reliable, fast, and easier. There are many reasons for the starting of the project because in the selling of items through the manual system of salesperson faces a lot of inefficiencies. It requires handling of large record books that consist of both irrelevant and important information's thus making it difficult to find out the required information as per necessity.

The administrators consist of a unique password and names of the employees. It helps the employees to make secure login. The ids and passwords are kept secret from others. The modules of sales and purchase include all the details of selling and purchasing. In the billing module the details of payments are clearly shown.

This is one of the best systems, that can be introduced in the supermarket for efficient management. It will also reduce the cumbersome job of finding the most accurate data from the huge logbooks. It also helps the management to keep efficient records of all the purchases and sales. The introduction of ID and password will further reduce the manipulation and thus providing the accurate and transparent data. This system will increase the productivity and reduce the need of manual system to a large extent.

Data Dictionary

User Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
username	Varchar			
password	varchar			
role	varchar			
name	varchar			
isactive	varchar			

Vendor Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
id	Integer			
vendor	varchar			
address	text			
contractperson	varchar			
telephone	varchar			
email	varchar			
fax	varchar			

Product Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
pcode	varchar			
barcode	varchar			
pdesc	varchar			
bid	Integer			
cid	Integer			
price	decimal			
qty	Integer			
reorder	Integer			

Stock Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
id	Integer			
refno	varchar			
pcode	varchar			
qty	Integer			
sdata	datetime			
stockinby	varchar			
status	varchar			
vendorid	Integer			

Categories Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
id	Integer			
category	varchar			

Cart Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
id	Integer			
transno	varchar			
pcode	varchar			
price	decimal			
qty	Integer			
disc precent	decimal			
disc	decimal			
total	decimal			
sdate	date			
status	varchar			
cashier	varchar			

Brand Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
id	Integer			
brand	varchar			

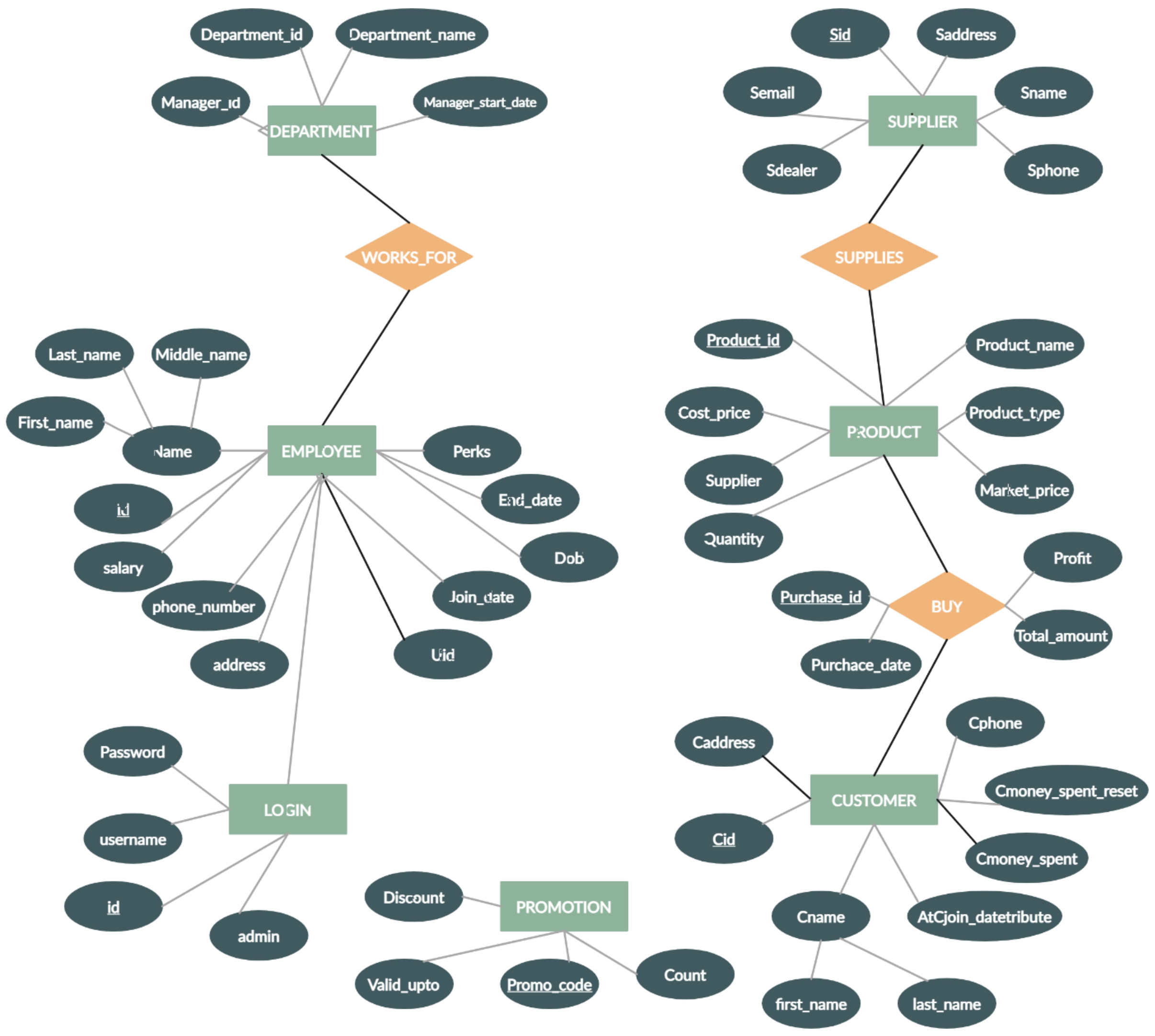
Adjustment Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
id	Integer			
referenceno	varchar			
pcode	varchar			
qty	Integer			
action	varchar			
remarks	text			
sdate	date			
user	varchar			

Cancel Table

Field Name	Data Type	Constraints	Constraint Name	Reference Table
id	Integer			
transno	varchar			
pcode	decimal			
price	Integer			
qty	Integer			
total	decimal			
sdate	date			
voidby	varchar			
cancelledby	varchar			
reason	text			
action	varchar			

ER Diagram

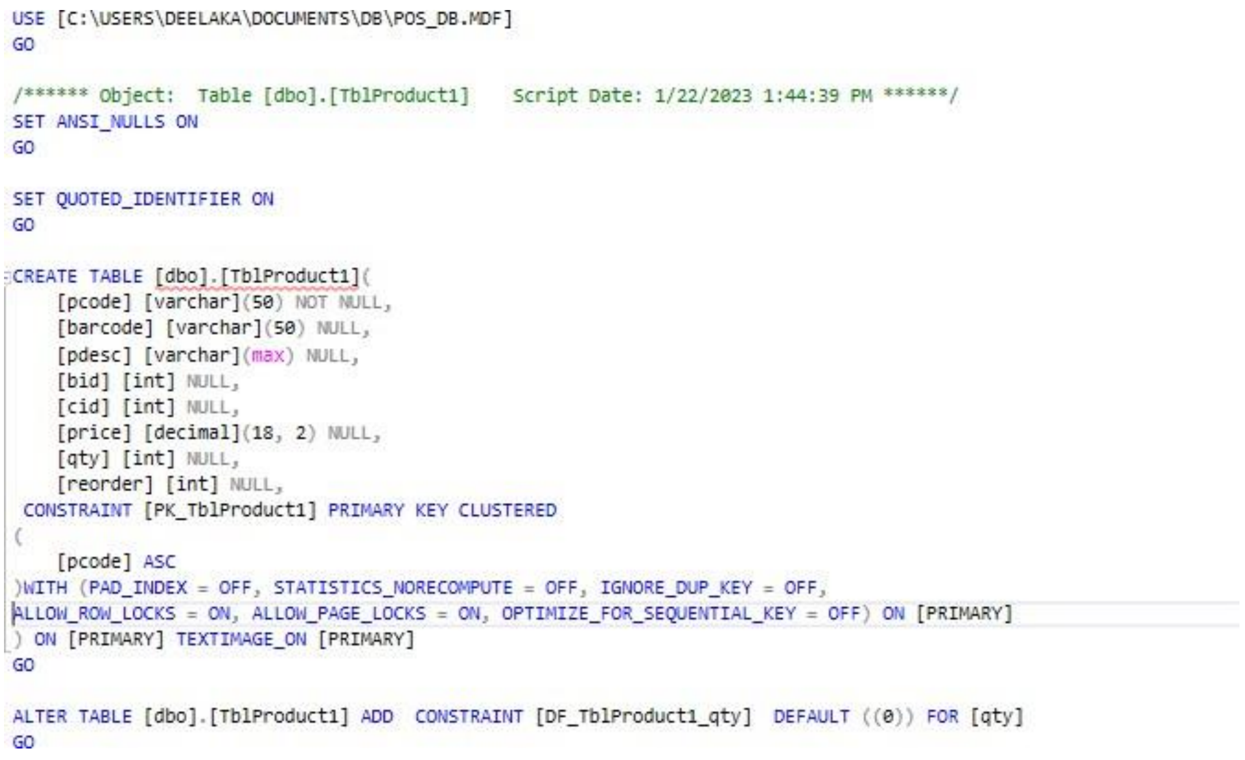


Department_id
Department_name
Manager_id
Manager_start_date
DEPARTMENT
WORKS_FOR
Last_name
Middle_name
First_name
Name
EMPLOYEE
Perks
End_date
Dob
Join_date
Uid
id
salary
phone_number
address
Password
LOGIN
username
id
admin
Discount
PROMOTION
Valid_upto
Promo_code
Count
Sid
Saddress
Semail
Sname
SUPPLIER
Sdealer
Sphone
SUPPLIES
Product_id
Product_name
Cost_price
PRODUCT
Product_type
Supplier
Market_price
Quantity
Profit
Purchase_id
BUY
Total_amount
Purchace_date
Cphone
Caddress
CUSTOMER
Cmoney_spent_reset
Cid
Cmoney_spent
Cname
AtCjoin_datetribute
first_name
last_name

Section 02

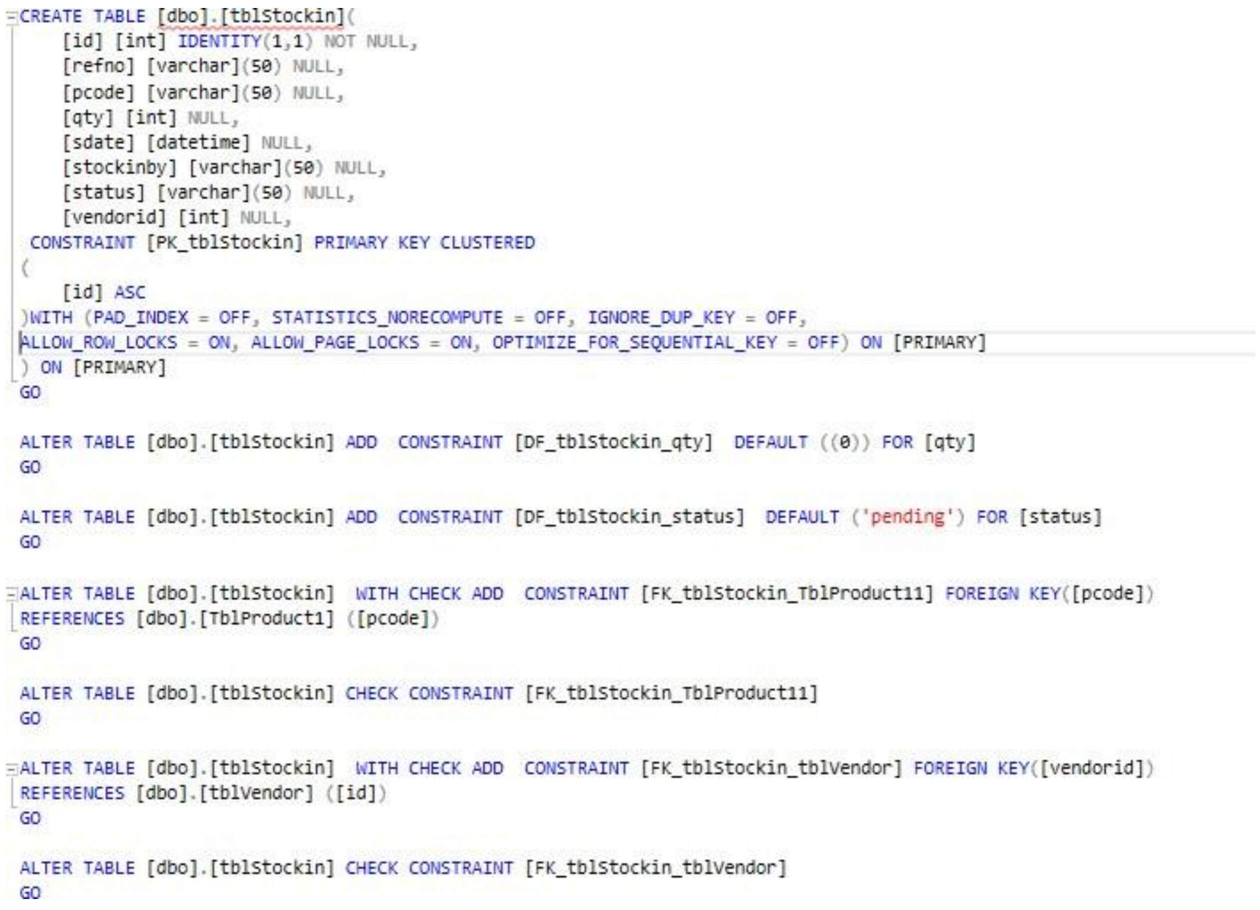
Create Table Statements

Product



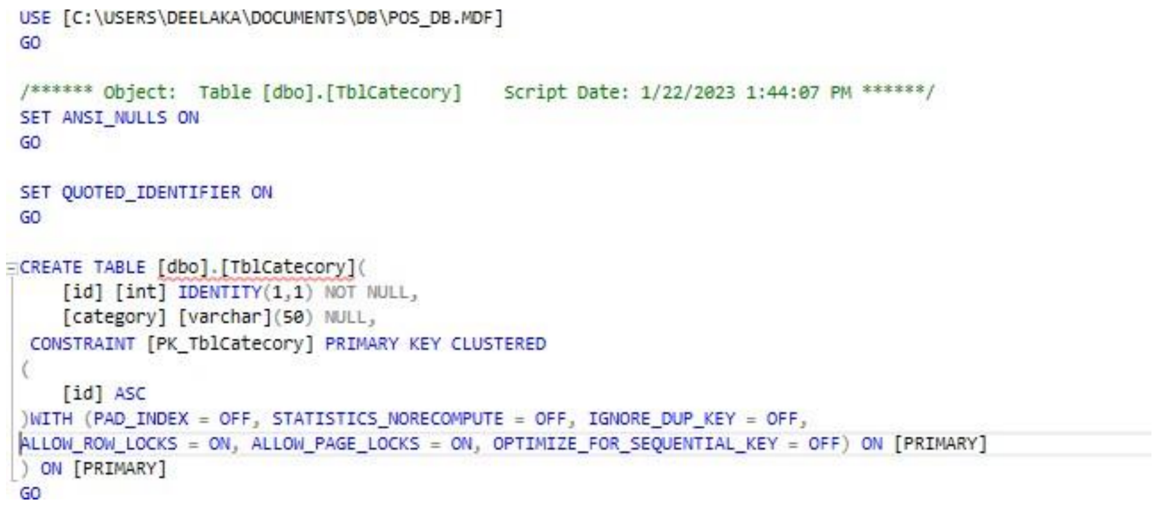
```
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[TblProduct1]    Script Date: 1/22/2023 1:44:39 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[TblProduct1](
    [pcode] [varchar](50) NOT NULL,
    [barcode] [varchar](50) NULL,
    [pdesc] [varchar](max) NULL,
    [bid] [int] NULL,
    [cid] [int] NULL,
    [price] [decimal](18, 2) NULL,
    [qty] [int] NULL,
    [reorder] [int] NULL,
 CONSTRAINT [PK_TblProduct1] PRIMARY KEY CLUSTERED
(
    [pcode] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO

ALTER TABLE [dbo].[TblProduct1] ADD  CONSTRAINT [DF_TblProduct1_qty]  DEFAULT ((0)) FOR [qty]
GO
```

Stock



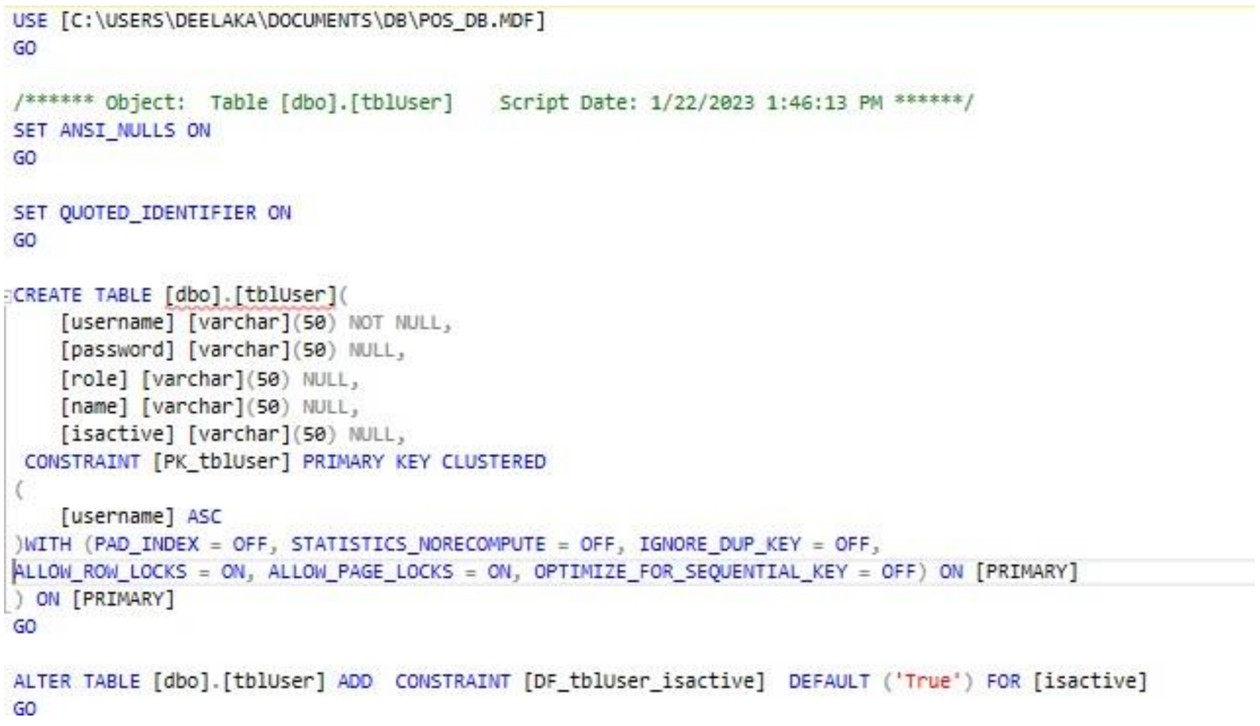
```
CREATE TABLE [dbo].[tblStockin](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [refno] [varchar](50) NULL,
    [pcode] [varchar](50) NULL,
    [qty] [int] NULL,
    [sdate] [datetime] NULL,
    [stockinby] [varchar](50) NULL,
    [status] [varchar](50) NULL,
    [vendorid] [int] NULL,
 CONSTRAINT [PK_tblStockin] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[tblStockin] ADD  CONSTRAINT [DF_tblStockin_qty]  DEFAULT ((0)) FOR [qty]
GO

ALTER TABLE [dbo].[tblStockin] ADD  CONSTRAINT [DF_tblStockin_status]  DEFAULT ('pending') FOR [status]
GO

ALTER TABLE [dbo].[tblStockin]  WITH CHECK ADD  CONSTRAINT [FK_tblStockin_TblProduct11] FOREIGN KEY([pcode])
REFERENCES [dbo].[TblProduct1] ([pcode])
GO

ALTER TABLE [dbo].[tblStockin] CHECK CONSTRAINT [FK_tblStockin_TblProduct11]
GO

ALTER TABLE [dbo].[tblStockin]  WITH CHECK ADD  CONSTRAINT [FK_tblStockin_tblVendor] FOREIGN KEY([vendorid])
REFERENCES [dbo].[tblVendor] ([id])
GO

ALTER TABLE [dbo].[tblStockin] CHECK CONSTRAINT [FK_tblStockin_tblVendor]
GO
```

Categories



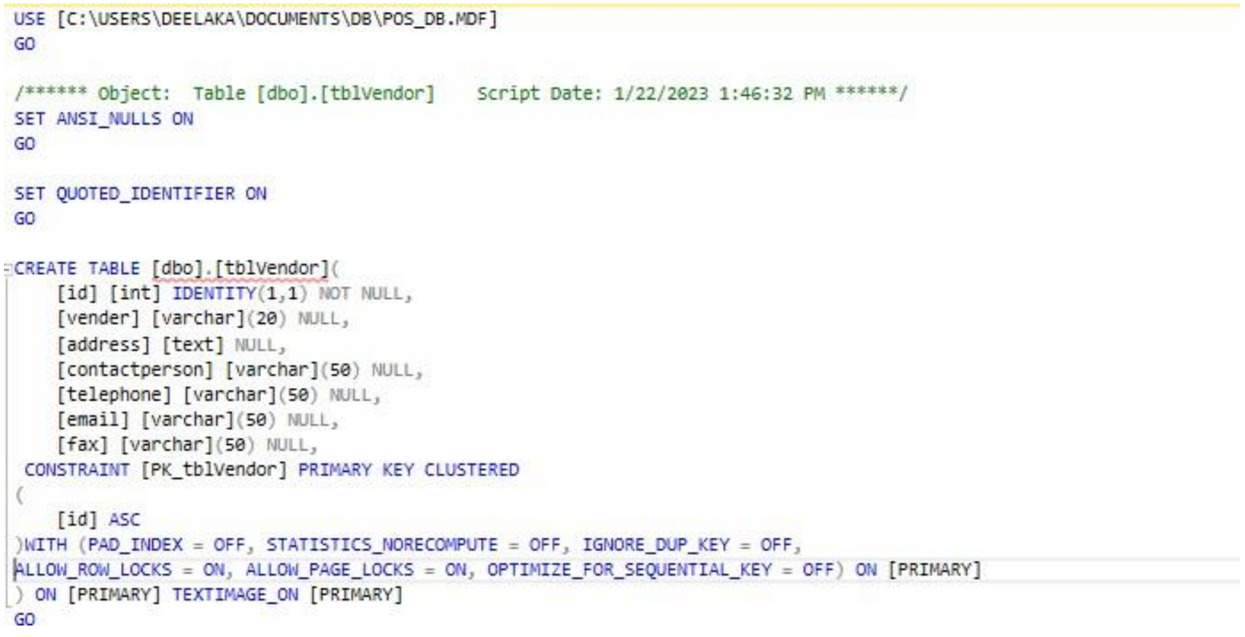
```
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[TblCatecory]    Script Date: 1/22/2023 1:44:07 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[TblCatecory](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [category] [varchar](50) NULL,
 CONSTRAINT [PK_TblCatecory] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO
```

User



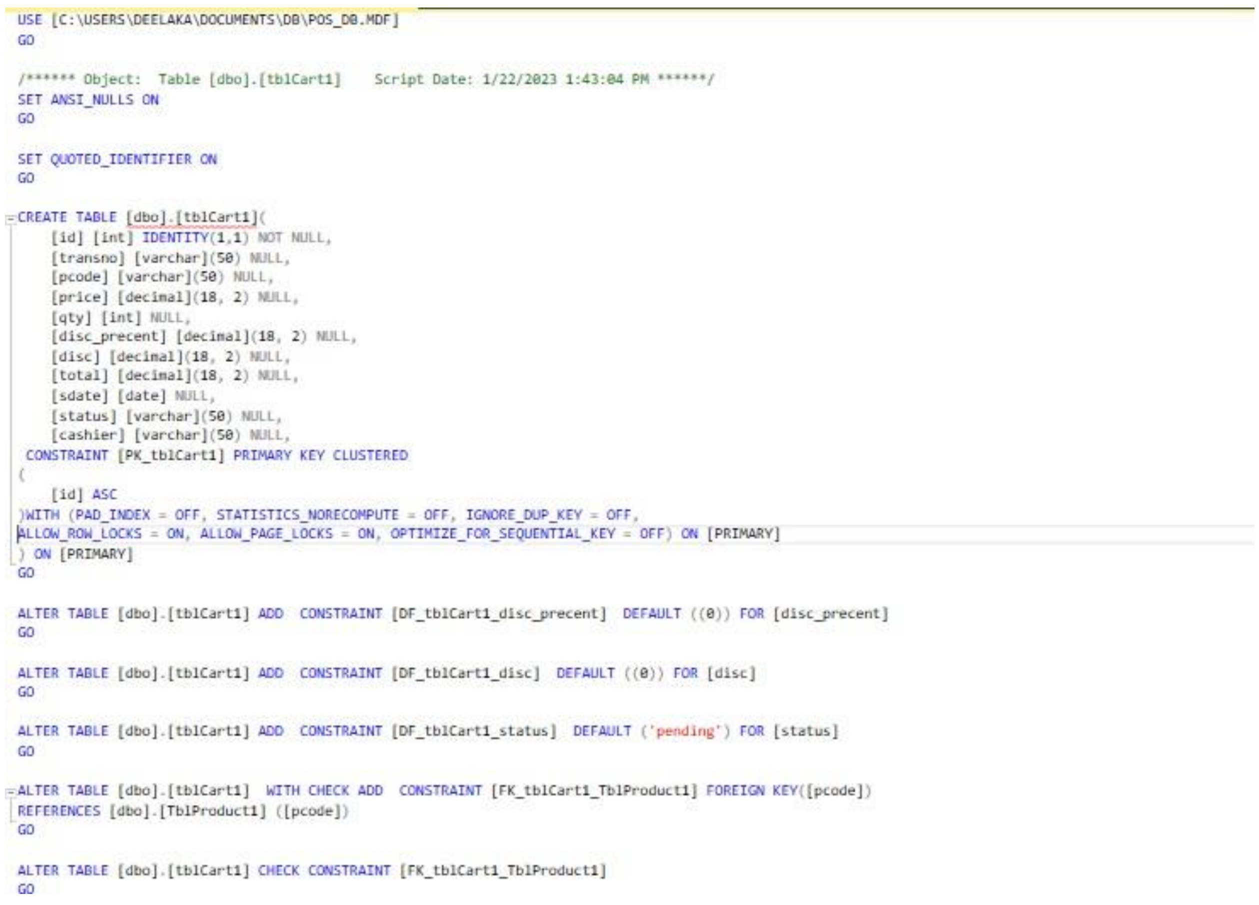
```
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[tblUser]    Script Date: 1/22/2023 1:46:13 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[tblUser](
    [username] [varchar](50) NOT NULL,
    [password] [varchar](50) NULL,
    [role] [varchar](50) NULL,
    [name] [varchar](50) NULL,
    [isactive] [varchar](50) NULL,
 CONSTRAINT [PK_tblUser] PRIMARY KEY CLUSTERED
(
    [username] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[tblUser] ADD  CONSTRAINT [DF_tblUser_isactive]  DEFAULT ('True') FOR [isactive]
GO
```

Vendor



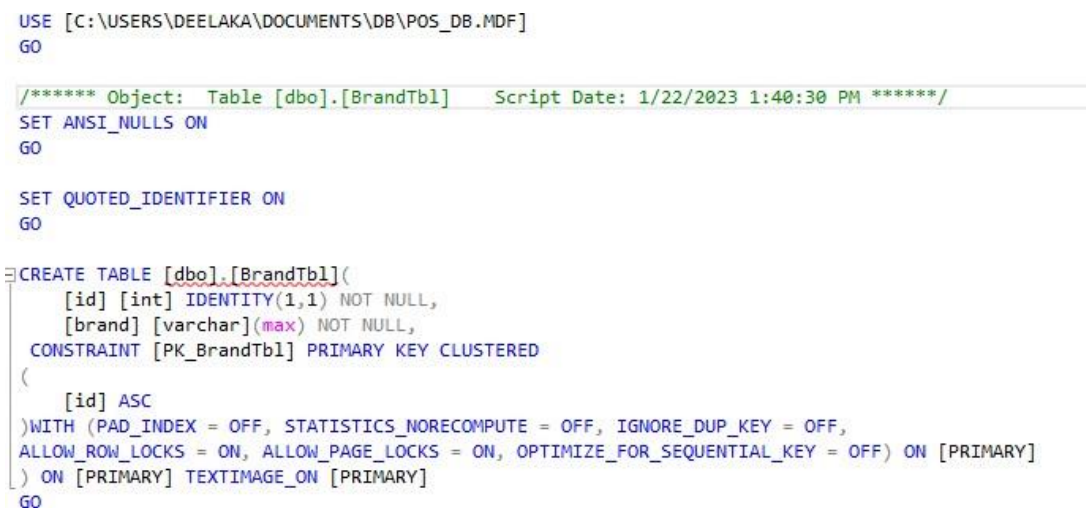
```
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[tblVendor]    Script Date: 1/22/2023 1:46:32 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[tblVendor](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [vender] [varchar](20) NULL,
    [address] [text] NULL,
    [contactperson] [varchar](50) NULL,
    [telephone] [varchar](50) NULL,
    [email] [varchar](50) NULL,
    [fax] [varchar](50) NULL,
 CONSTRAINT [PK_tblVendor] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
```

Cart



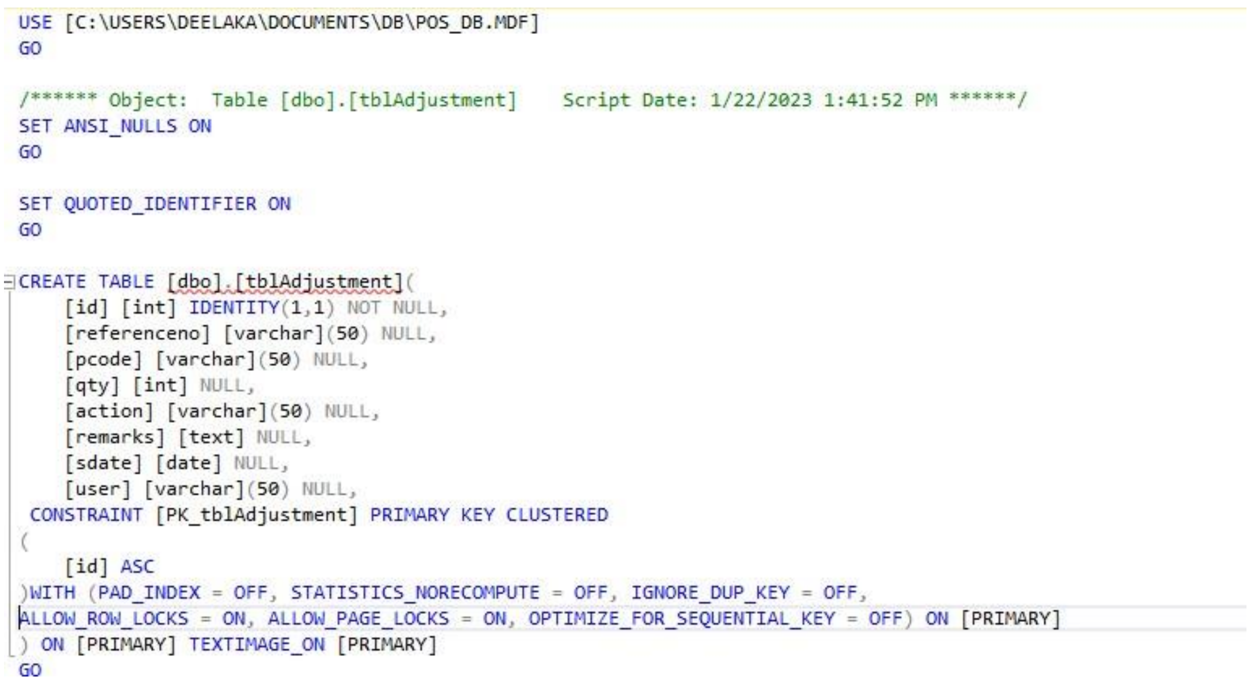
```
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[tblCart1]    Script Date: 1/22/2023 1:43:04 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[tblCart1](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [transno] [varchar](50) NULL,
    [pcode] [varchar](50) NULL,
    [price] [decimal](18, 2) NULL,
    [qty] [int] NULL,
    [disc_precent] [decimal](18, 2) NULL,
    [disc] [decimal](18, 2) NULL,
    [total] [decimal](18, 2) NULL,
    [sdate] [date] NULL,
    [status] [varchar](50) NULL,
    [cashier] [varchar](50) NULL,
 CONSTRAINT [PK_tblCart1] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[tblCart1] ADD  CONSTRAINT [DF_tblCart1_disc_precent]  DEFAULT ((0)) FOR [disc_precent]
GO

ALTER TABLE [dbo].[tblCart1] ADD  CONSTRAINT [DF_tblCart1_disc]  DEFAULT ((0)) FOR [disc]
GO

ALTER TABLE [dbo].[tblCart1] ADD  CONSTRAINT [DF_tblCart1_status]  DEFAULT ('pending') FOR [status]
GO

ALTER TABLE [dbo].[tblCart1]  WITH CHECK ADD  CONSTRAINT [FK_tblCart1_TblProduct1] FOREIGN KEY([pcode])
REFERENCES [dbo].[TblProduct1] ([pcode])
GO

ALTER TABLE [dbo].[tblCart1] CHECK CONSTRAINT [FK_tblCart1_TblProduct1]
GO
```

Brand



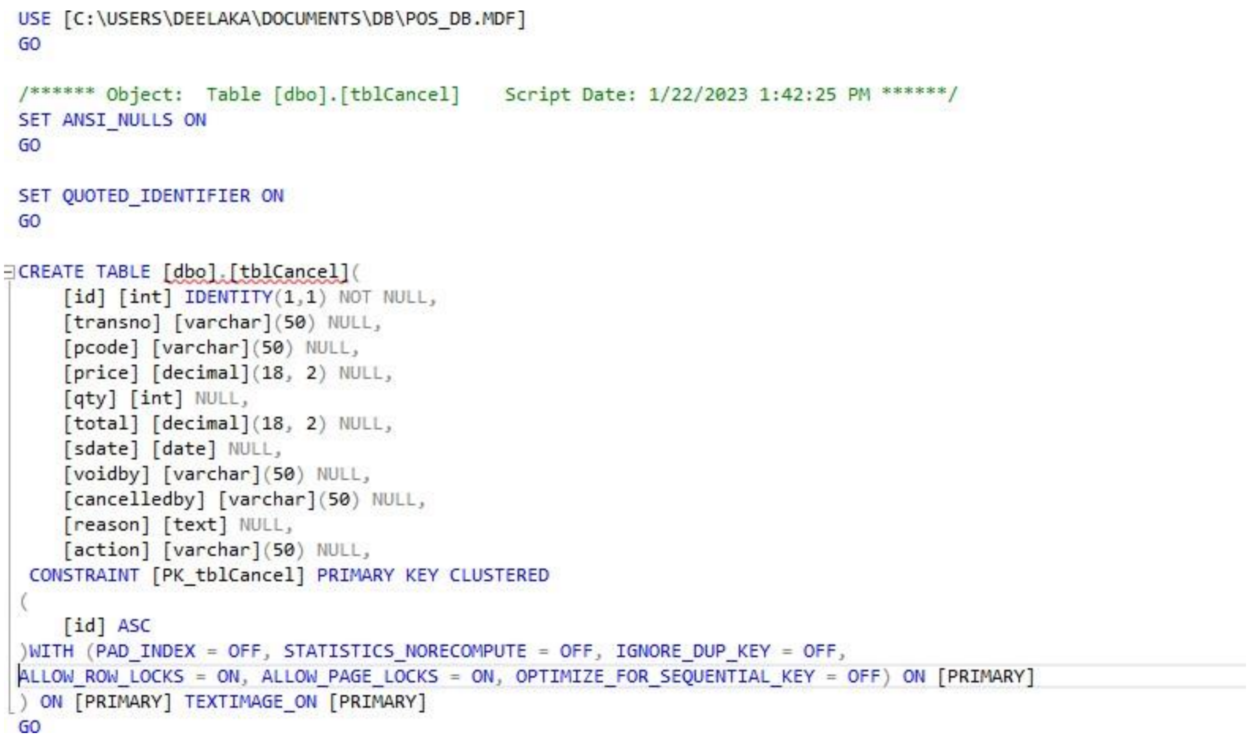
```
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[BrandTbl]    Script Date: 1/22/2023 1:40:30 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[BrandTbl](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [brand] [varchar](max) NOT NULL,
 CONSTRAINT [PK_BrandTbl] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
```

Adjustment



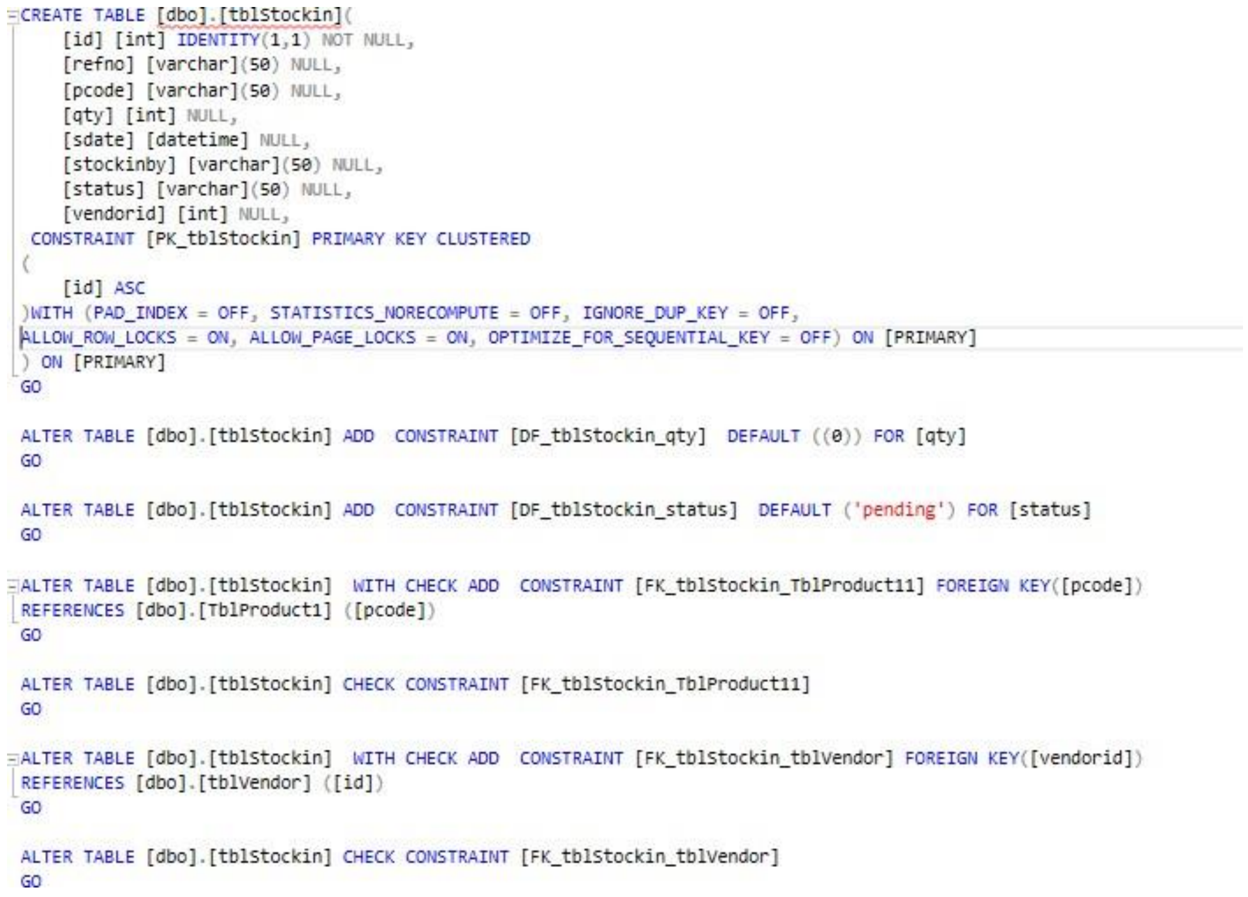
```sql
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[tblAdjustment]    Script Date: 1/22/2023 1:41:52 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[tblAdjustment](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [referenceno] [varchar](50) NULL,
    [pcode] [varchar](50) NULL,
    [qty] [int] NULL,
    [action] [varchar](50) NULL,
    [remarks] [text] NULL,
    [sdate] [date] NULL,
    [user] [varchar](50) NULL,
 CONSTRAINT [PK_tblAdjustment] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
```

Cancel



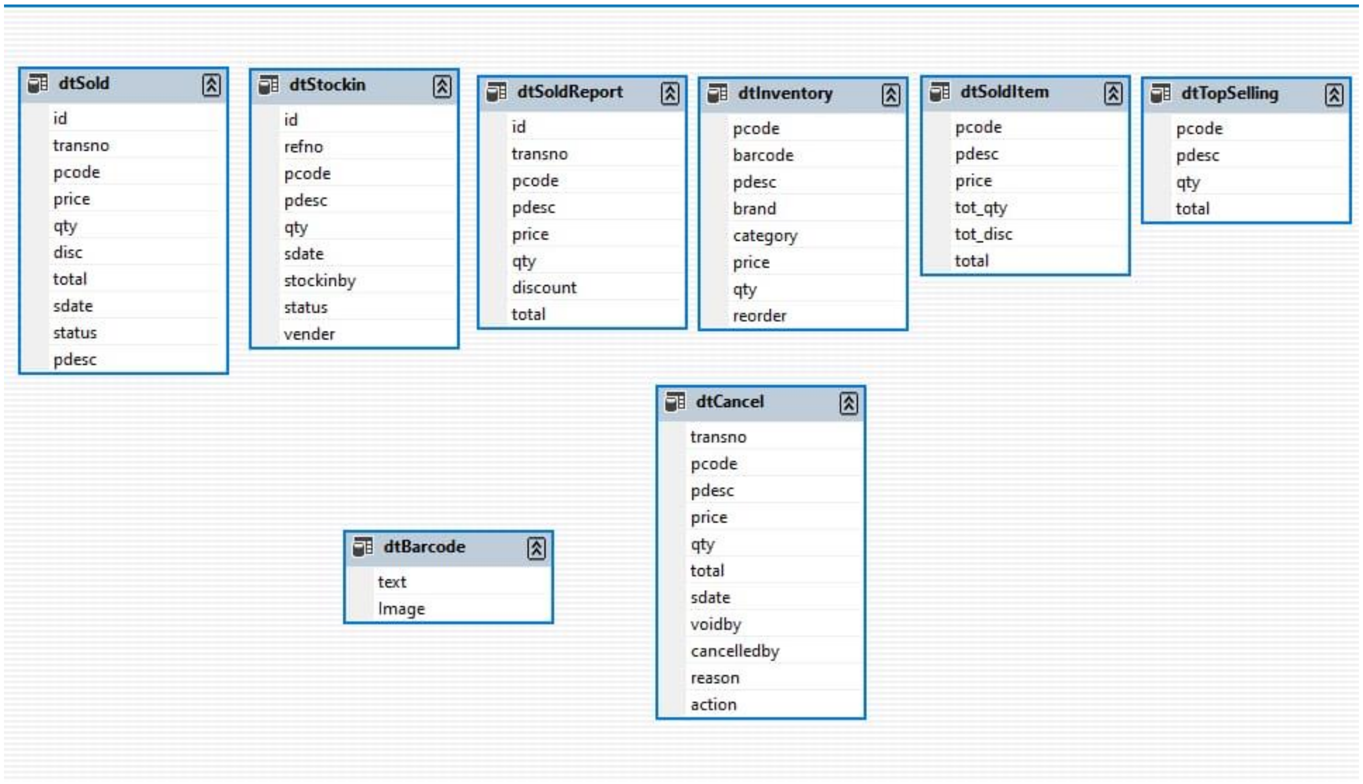
```sql
USE [C:\USERS\DEELAKA\DOCUMENTS\DB\POS_DB.MDF]
GO

/****** Object:  Table [dbo].[tblCancel]    Script Date: 1/22/2023 1:42:25 PM ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[tblCancel](
    [id] [int] IDENTITY(1,1) NOT NULL,
    [transno] [varchar](50) NULL,
    [pcode] [varchar](50) NULL,
    [price] [decimal](18, 2) NULL,
    [qty] [int] NULL,
    [total] [decimal](18, 2) NULL,
    [sdate] [date] NULL,
    [voidby] [varchar](50) NULL,
    [cancelledby] [varchar](50) NULL,
    [reason] [text] NULL,
    [action] [varchar](50) NULL,
 CONSTRAINT [PK_tblCancel] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
```

Stock



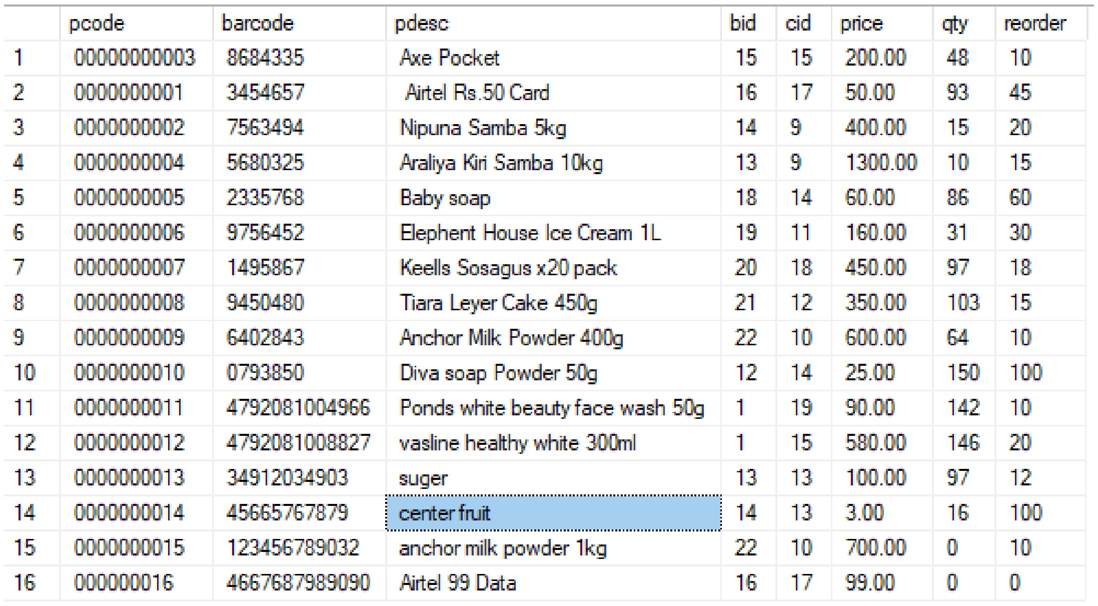
```
CREATE TABLE [dbo].[tblStockin](
	[id] [int] IDENTITY(1,1) NOT NULL,
	[refno] [varchar](50) NULL,
	[pcode] [varchar](50) NULL,
	[qty] [int] NULL,
	[sdate] [datetime] NULL,
	[stockinby] [varchar](50) NULL,
	[status] [varchar](50) NULL,
	[vendorid] [int] NULL,
 CONSTRAINT [PK_tblStockin] PRIMARY KEY CLUSTERED
(
	[id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON, OPTIMIZE_FOR_SEQUENTIAL_KEY = OFF) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[tblStockin] ADD  CONSTRAINT [DF_tblStockin_qty]  DEFAULT ((0)) FOR [qty]
GO

ALTER TABLE [dbo].[tblStockin] ADD  CONSTRAINT [DF_tblStockin_status]  DEFAULT ('pending') FOR [status]
GO

ALTER TABLE [dbo].[tblStockin]  WITH CHECK ADD  CONSTRAINT [FK_tblStockin_TblProduct11] FOREIGN KEY([pcode])
REFERENCES [dbo].[TblProduct1] ([pcode])
GO

ALTER TABLE [dbo].[tblStockin] CHECK CONSTRAINT [FK_tblStockin_TblProduct11]
GO

ALTER TABLE [dbo].[tblStockin]  WITH CHECK ADD  CONSTRAINT [FK_tblStockin_tblVendor] FOREIGN KEY([vendorid])
REFERENCES [dbo].[tblVendor] ([id])
GO

ALTER TABLE [dbo].[tblStockin] CHECK CONSTRAINT [FK_tblStockin_tblVendor]
GO
```

Database Diagram



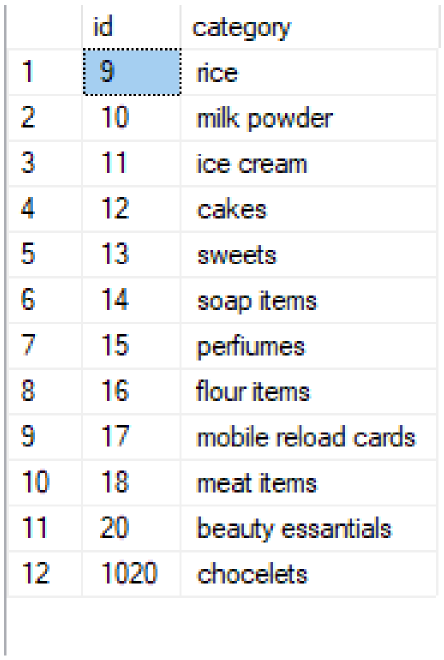
Sample Records

Product Table



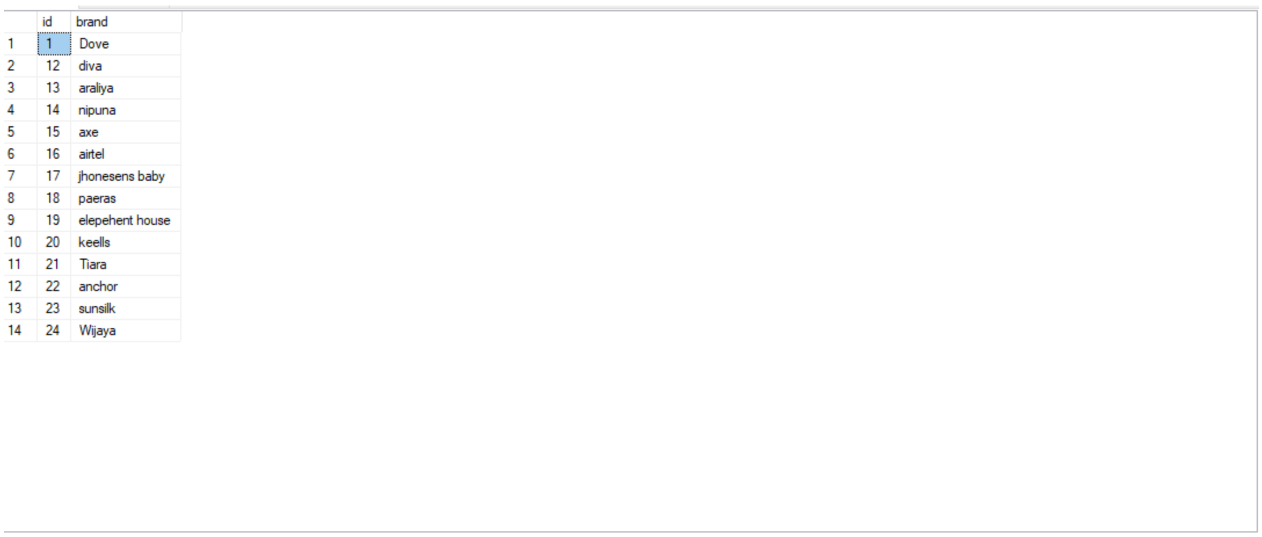
	pcode	barcode	pdesc	bid	cid	price	qty	reorder
1	00000000003	8684335	Axe Pocket	15	15	200.00	48	10
2	0000000001	3454657	Airtel Rs.50 Card	16	17	50.00	93	45
3	0000000002	7563494	Nipuna Samba 5kg	14	9	400.00	15	20
4	0000000004	5680325	Araliya Kiri Samba 10kg	13	9	1300.00	10	15
5	0000000005	2335768	Baby soap	18	14	60.00	86	60
6	0000000006	9756452	Elephent House Ice Cream 1L	19	11	160.00	31	30
7	0000000007	1495867	Keells Sosagus x20 pack	20	18	450.00	97	18
8	0000000008	9450480	Tiara Leyer Cake 450g	21	12	350.00	103	15
9	0000000009	6402843	Anchor Milk Powder 400g	22	10	600.00	64	10
10	0000000010	0793850	Diva soap Powder 50g	12	14	25.00	150	100
11	0000000011	4792081004966	Ponds white beauty face wash 50g	1	19	90.00	142	10
12	0000000012	4792081008827	vasline healthy white 300ml	1	15	580.00	146	20
13	0000000013	34912034903	suger	13	13	100.00	97	12
14	0000000014	45665767879	center fruit	14	13	3.00	16	100
15	0000000015	123456789032	anchor milk powder 1kg	22	10	700.00	0	10
16	000000016	4667687989090	Airtel 99 Data	16	17	99.00	0	0

Categories Table



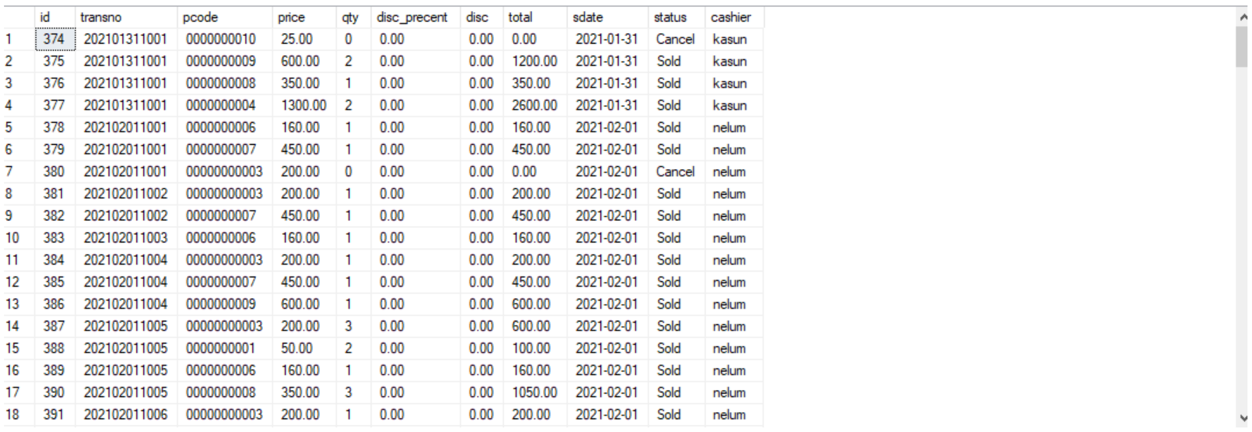
	id	category
1	9	rice
2	10	milk powder
3	11	ice cream
4	12	cakes
5	13	sweets
6	14	soap items
7	15	perfiumes
8	16	flour items
9	17	mobile reload cards
10	18	meat items
11	20	beauty essantials
12	1020	chocelets

Brands Table



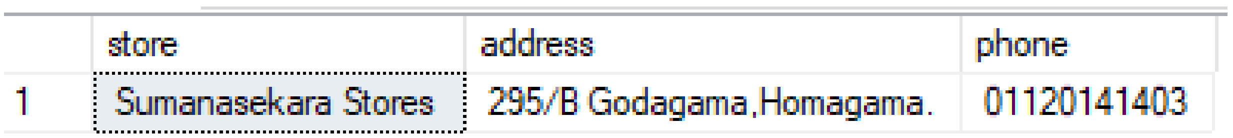
	id	brand
1	1	Dove
2	12	diva
3	13	araliya
4	14	nipuna
5	15	axe
6	16	airtel
7	17	jhonesens baby
8	18	paeras
9	19	elepehent house
10	20	keells
11	21	Tiara
12	22	anchor
13	23	sunsilk
14	24	Wijaya

Cart Table



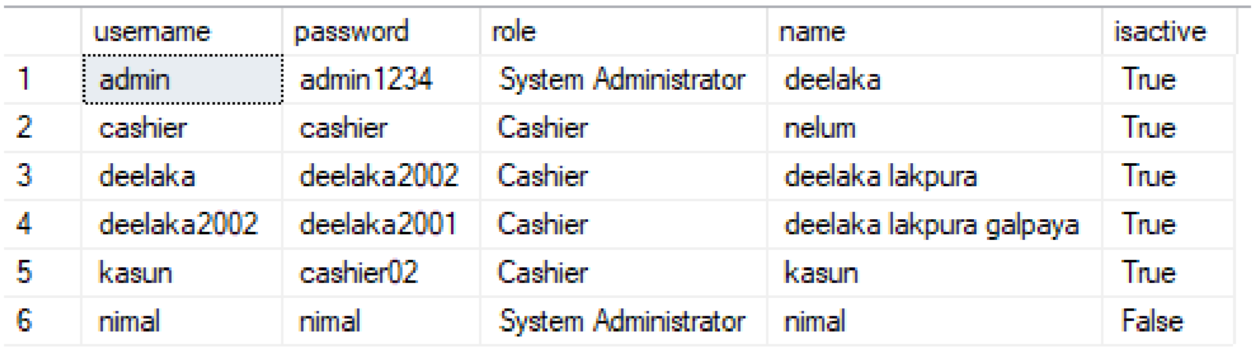
	id	transno	pcode	price	qty	disc_precent	disc	total	sdate	status	cashier
1	374	202101311001	0000000010	25.00	0	0.00	0.00	0.00	2021-01-31	Cancel	kasun
2	375	202101311001	0000000009	600.00	2	0.00	0.00	1200.00	2021-01-31	Sold	kasun
3	376	202101311001	0000000008	350.00	1	0.00	0.00	350.00	2021-01-31	Sold	kasun
4	377	202101311001	0000000004	1300.00	2	0.00	0.00	2600.00	2021-01-31	Sold	kasun
5	378	202102011001	0000000006	160.00	1	0.00	0.00	160.00	2021-02-01	Sold	nelum
6	379	202102011001	0000000007	450.00	1	0.00	0.00	450.00	2021-02-01	Sold	nelum
7	380	202102011001	00000000003	200.00	0	0.00	0.00	0.00	2021-02-01	Cancel	nelum
8	381	202102011002	00000000003	200.00	1	0.00	0.00	200.00	2021-02-01	Sold	nelum
9	382	202102011002	0000000007	450.00	1	0.00	0.00	450.00	2021-02-01	Sold	nelum
10	383	202102011003	0000000006	160.00	1	0.00	0.00	160.00	2021-02-01	Sold	nelum
11	384	202102011004	00000000003	200.00	1	0.00	0.00	200.00	2021-02-01	Sold	nelum
12	385	202102011004	0000000007	450.00	1	0.00	0.00	450.00	2021-02-01	Sold	nelum
13	386	202102011004	0000000009	600.00	1	0.00	0.00	600.00	2021-02-01	Sold	nelum
14	387	202102011005	00000000003	200.00	3	0.00	0.00	600.00	2021-02-01	Sold	nelum
15	388	202102011005	0000000001	50.00	2	0.00	0.00	100.00	2021-02-01	Sold	nelum
16	389	202102011005	0000000006	160.00	1	0.00	0.00	160.00	2021-02-01	Sold	nelum
17	390	202102011005	0000000008	350.00	3	0.00	0.00	1050.00	2021-02-01	Sold	nelum
18	391	202102011006	00000000003	200.00	1	0.00	0.00	200.00	2021-02-01	Sold	nelum

Store Table



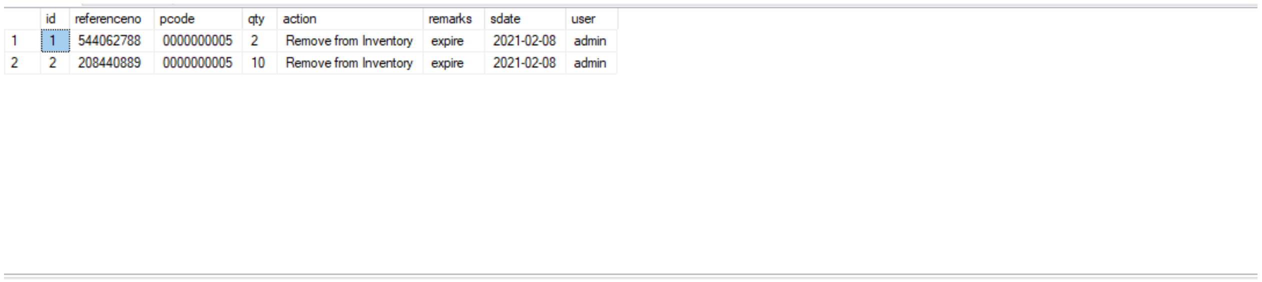
	store	address	phone
1	Sumanasekara Stores	295/B Godagama,Homagama.	01120141403

User Table



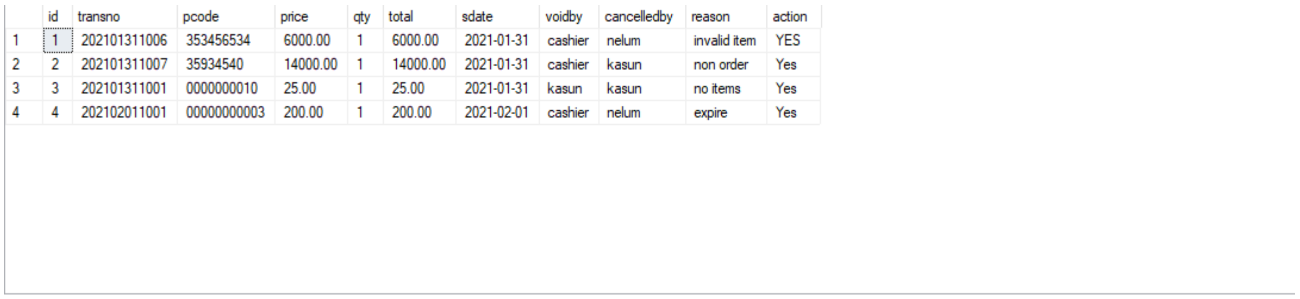
	username	password	role	name	isactive
1	admin	admin1234	System Administrator	deelaka	True
2	cashier	cashier	Cashier	nelum	True
3	deelaka	deelaka2002	Cashier	deelaka lakpura	True
4	deelaka2002	deelaka2001	Cashier	deelaka lakpura galpaya	True
5	kasun	cashier02	Cashier	kasun	True
6	nimal	nimal	System Administrator	nimal	False

Adjustment Table



	id	referenceno	pcode	qty	action	remarks	sdate	user
1	1	544062788	0000000005	2	Remove from Inventory	expire	2021-02-08	admin
2	2	208440889	0000000005	10	Remove from Inventory	expire	2021-02-08	admin

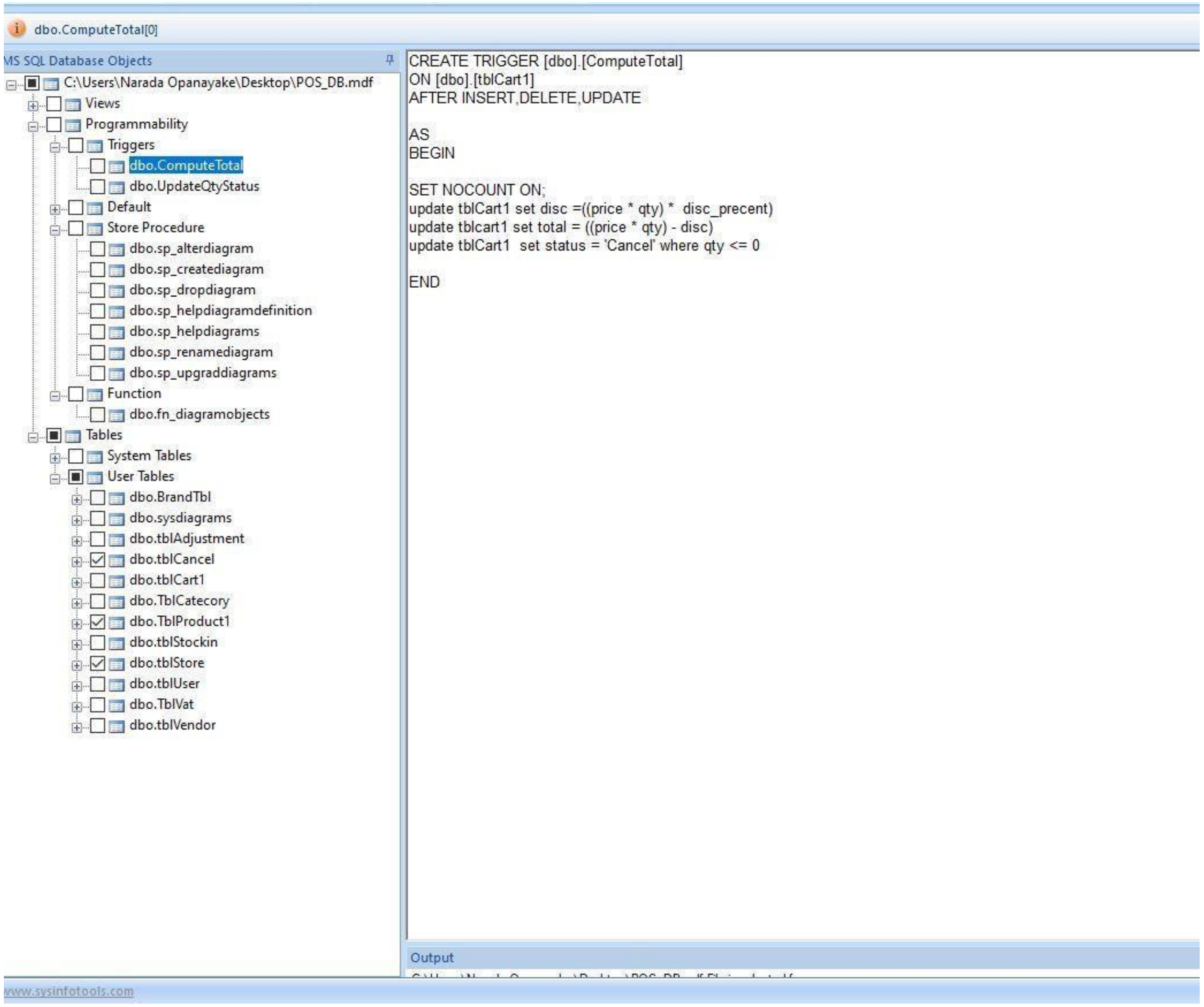
Oder Cancel Table



	id	transno	pcode	price	qty	total	sdate	voidby	cancelledby	reason	action
1	1	202101311006	353456534	6000.00	1	6000.00	2021-01-31	cashier	nelum	invalid item	YES
2	2	202101311007	35934540	14000.00	1	14000.00	2021-01-31	cashier	kasun	non order	Yes
3	3	202101311001	0000000010	25.00	1	25.00	2021-01-31	kasun	kasun	no items	Yes
4	4	202102011001	00000000003	200.00	1	200.00	2021-02-01	cashier	nelum	expire	Yes

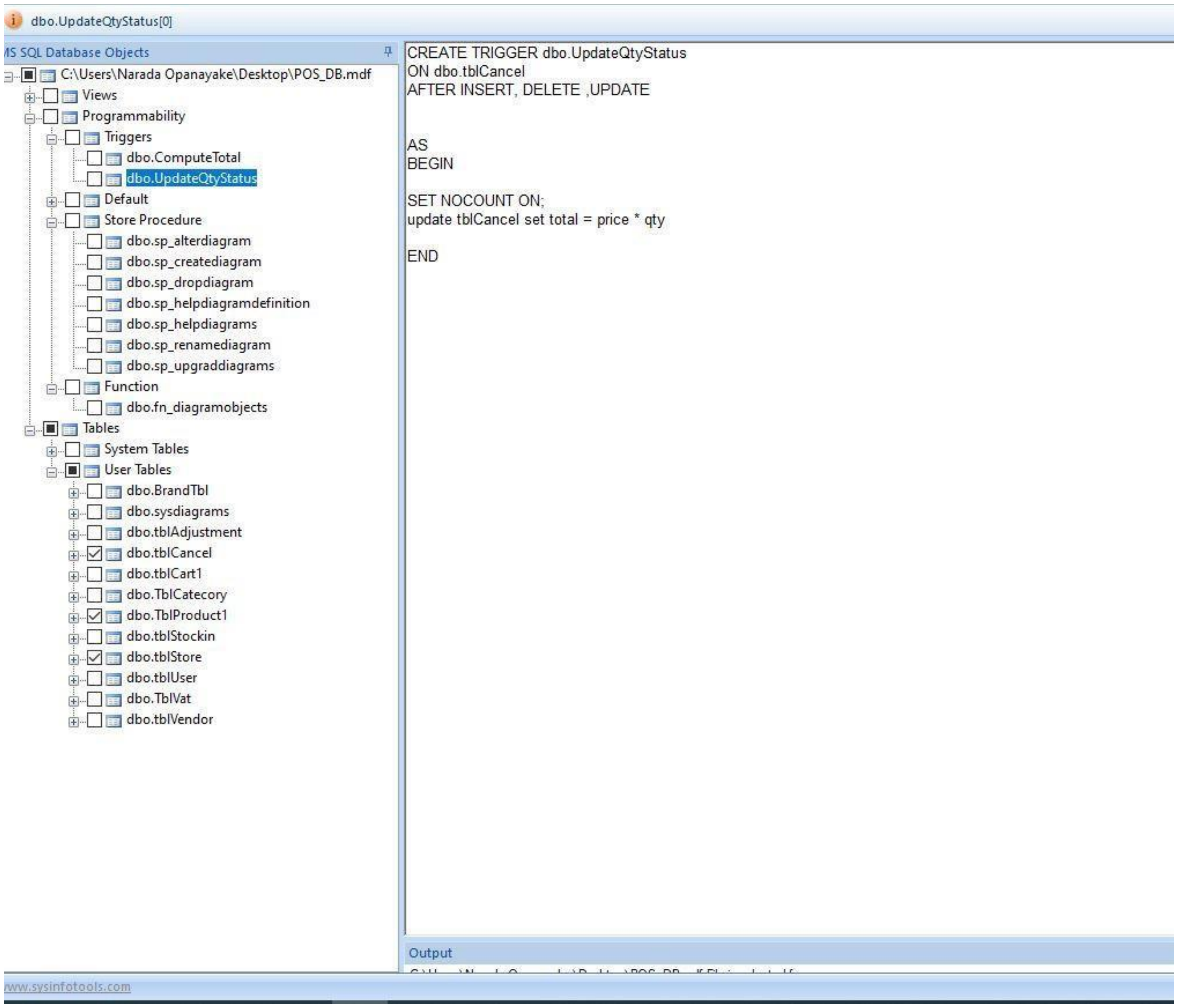
Section 03

Create Trigger Statements



```
CREATE TRIGGER [dbo].[ComputeTotal]
ON [dbo].[tblCart1]
AFTER INSERT,DELETE,UPDATE

AS
BEGIN

SET NOCOUNT ON;
update tblCart1 set disc =((price * qty) *  disc_precent)
update tblcart1 set total = ((price * qty) - disc)
update tblCart1  set status = 'Cancel' where qty <= 0

END
```

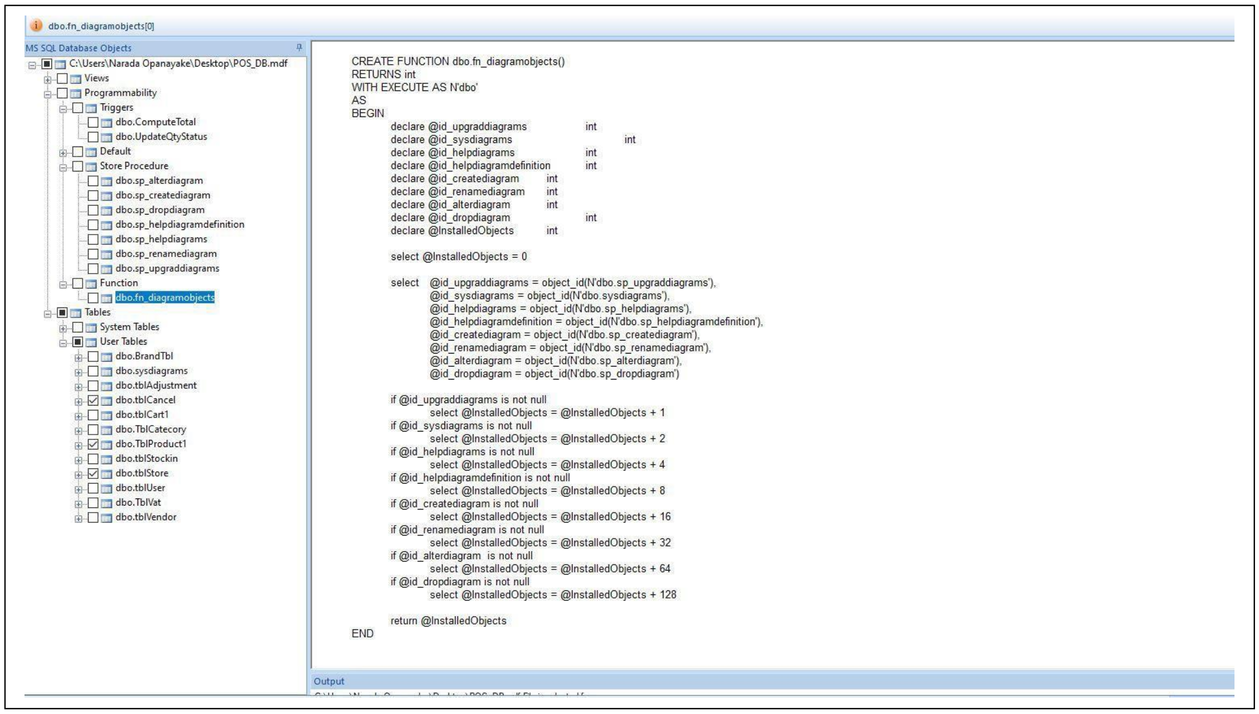
Trigger 1



```
CREATE TRIGGER dbo.UpdateQtyStatus
ON dbo.tblCancel
AFTER INSERT, DELETE ,UPDATE


AS
BEGIN

SET NOCOUNT ON;
update tblCancel set total = price * qty

END
```

Trigger 2

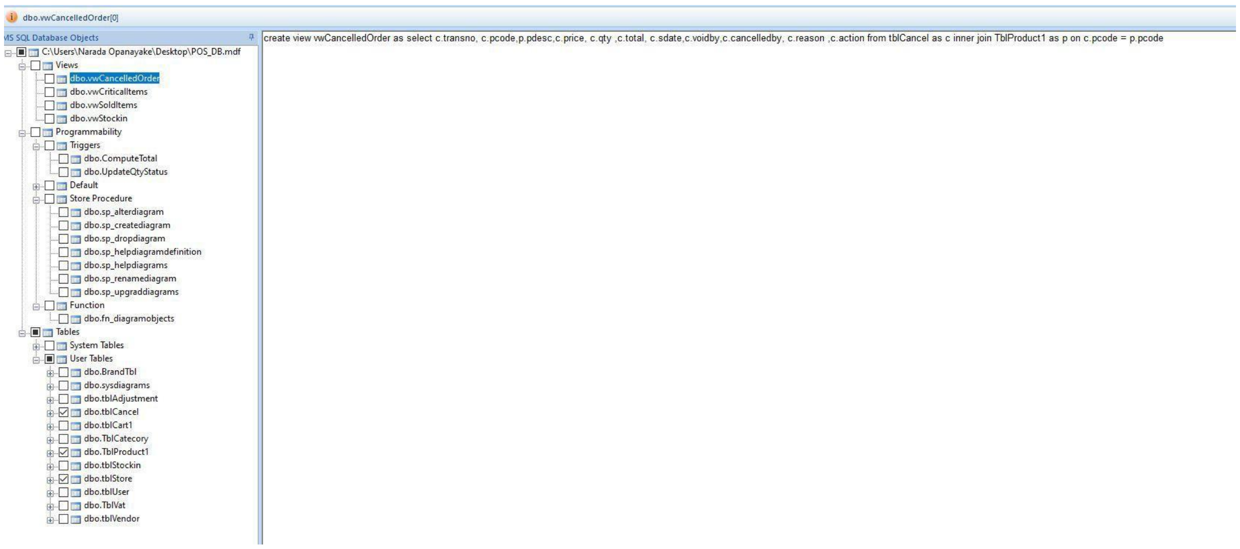
Create Function Statements



```
CREATE FUNCTION dbo.fn_diagramobjects()
RETURNS int
WITH EXECUTE AS N'dbo'
AS
BEGIN
          declare @id_upgraddiagrams              int
          declare @id_sysdiagrams                           int
          declare @id_helpdiagrams                int
          declare @id_helpdiagramdefinition       int
          declare @id_creatediagram       int
          declare @id_renamediagram       int
          declare @id_alterdiagram        int
          declare @id_dropdiagram                 int
          declare @InstalledObjects       int

          select @InstalledObjects = 0

          select   @id_upgraddiagrams = object_id(N'dbo.sp_upgraddiagrams'),
                   @id_sysdiagrams = object_id(N'dbo.sysdiagrams'),
                   @id_helpdiagrams = object_id(N'dbo.sp_helpdiagrams'),
                   @id_helpdiagramdefinition = object_id(N'dbo.sp_helpdiagramdefinition'),
                   @id_creatediagram = object_id(N'dbo.sp_creatediagram'),
                   @id_renamediagram = object_id(N'dbo.sp_renamediagram'),
                   @id_alterdiagram = object_id(N'dbo.sp_alterdiagram'),
                   @id_dropdiagram = object_id(N'dbo.sp_dropdiagram')

          if @id_upgraddiagrams is not null
                   select @InstalledObjects = @InstalledObjects + 1
          if @id_sysdiagrams is not null
                   select @InstalledObjects = @InstalledObjects + 2
          if @id_helpdiagrams is not null
                   select @InstalledObjects = @InstalledObjects + 4
          if @id_helpdiagramdefinition is not null
                   select @InstalledObjects = @InstalledObjects + 8
          if @id_creatediagram is not null
                   select @InstalledObjects = @InstalledObjects + 16
          if @id_renamediagram is not null
                   select @InstalledObjects = @InstalledObjects + 32
          if @id_alterdiagram  is not null
                   select @InstalledObjects = @InstalledObjects + 64
          if @id_dropdiagram is not null
                   select @InstalledObjects = @InstalledObjects + 128

          return @InstalledObjects
END
```

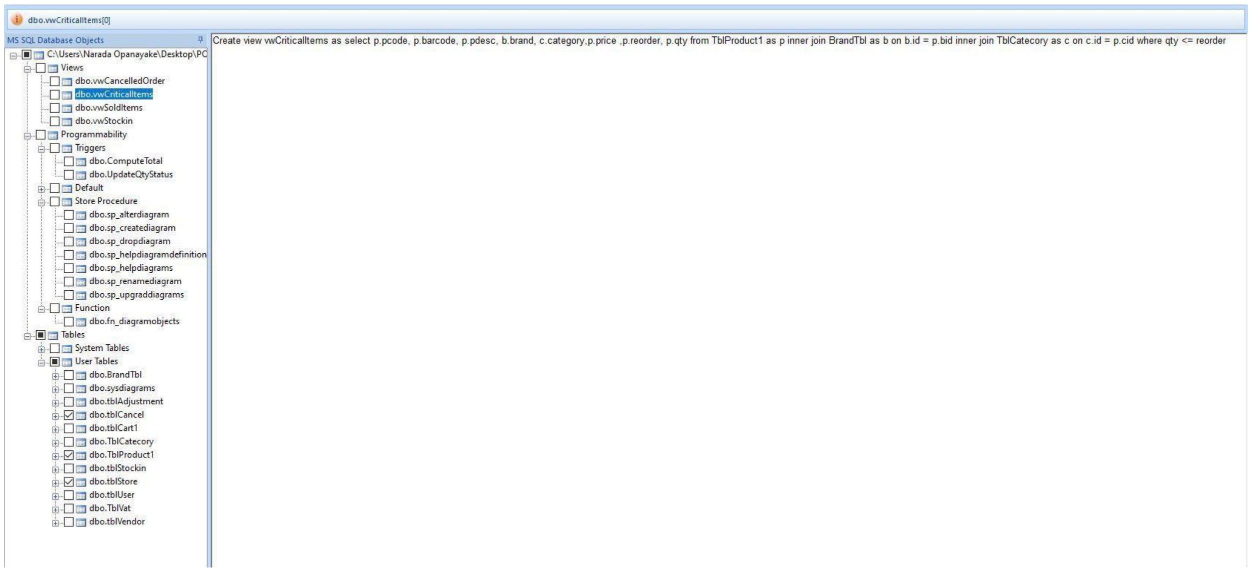
Function 1

Create View Statements



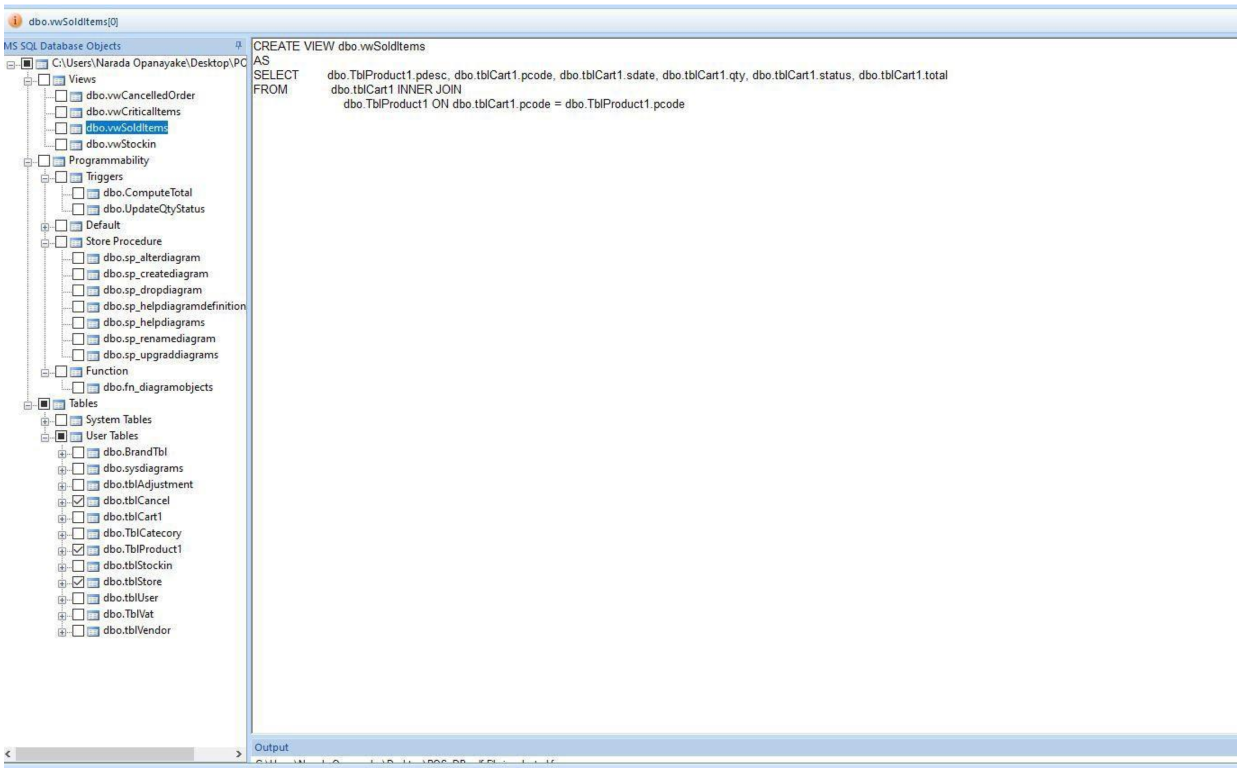
```
create view vwCancelledOrder as select c.transno, c.pcode,p.pdesc,c.price, c.qty ,c.total, c.sdate,c.voidby,c.cancelledby, c.reason ,c.action from tblCancel as c inner join TblProduct1 as p on c.pcode = p.pcode
```

View 1



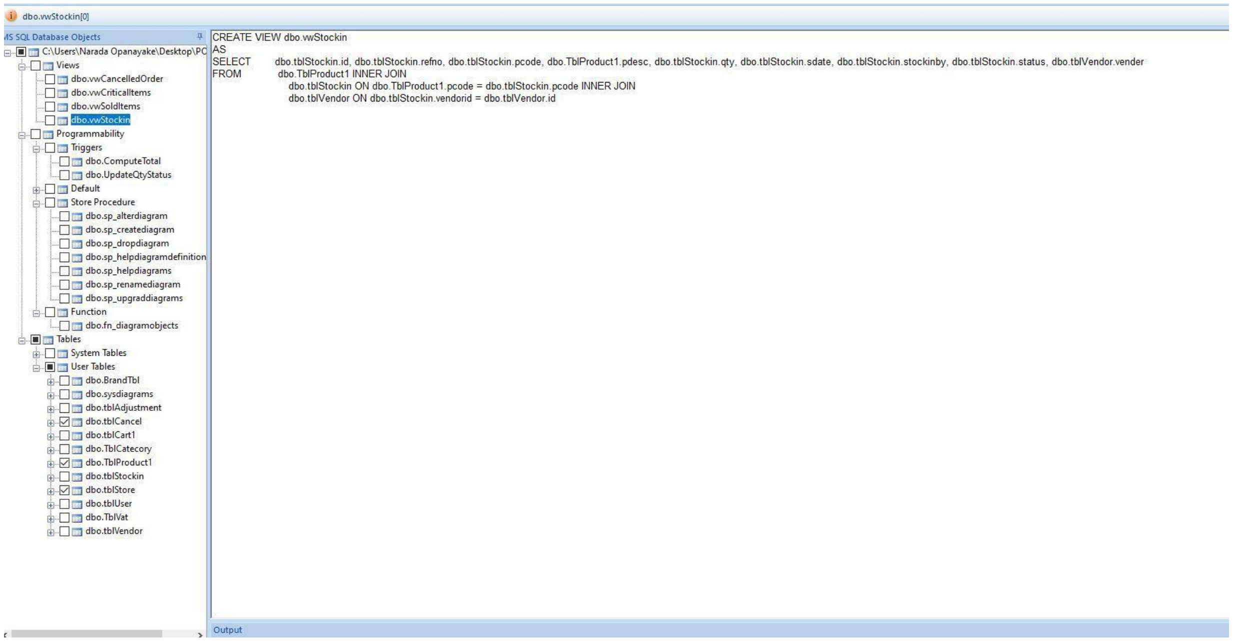
```
Create view vwCriticalItems as select p.pcode, p.barcode, p.pdesc, b.brand, c.category,p.price ,p.reorder, p.qty from TblProduct1 as p inner join BrandTbl as b on b.id = p.bid inner join TblCatecory as c on c.id = p.cid where qty <= reorder
```

View 2



```sql
CREATE VIEW dbo.vwSoldItems
AS
SELECT      dbo.TblProduct1.pdesc, dbo.tblCart1.pcode, dbo.tblCart1.sdate, dbo.tblCart1.qty, dbo.tblCart1.status, dbo.tblCart1.total
FROM         dbo.tblCart1 INNER JOIN
                 dbo.TblProduct1 ON dbo.tblCart1.pcode = dbo.TblProduct1.pcode
```

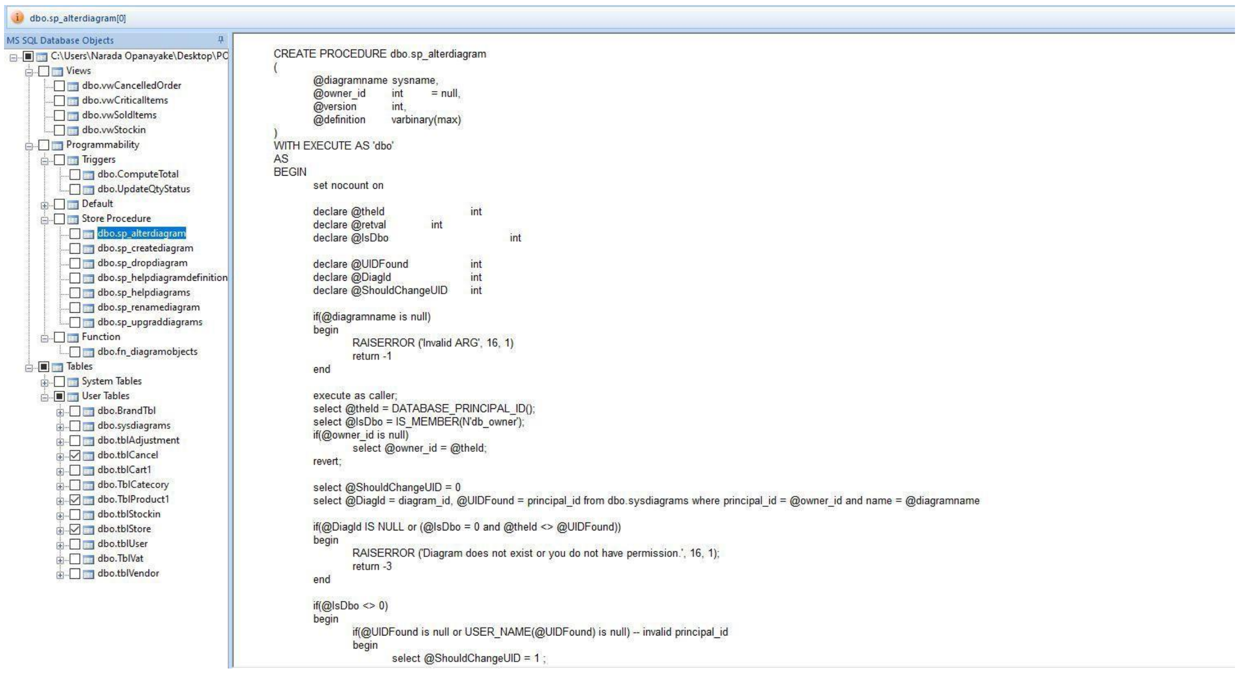
View 3



```sql
CREATE VIEW dbo.vwStockin
AS
SELECT      dbo.tblStockin.id, dbo.tblStockin.refno, dbo.tblStockin.pcode, dbo.TblProduct1.pdesc, dbo.tblStockin.qty, dbo.tblStockin.sdate, dbo.tblStockin.stockinby, dbo.tblStockin.status, dbo.tblVendor.vender
FROM         dbo.TblProduct1 INNER JOIN
                 dbo.tblStockin ON dbo.TblProduct1.pcode = dbo.tblStockin.pcode INNER JOIN
                 dbo.tblVendor ON dbo.tblStockin.vendorid = dbo.tblVendor.id
```

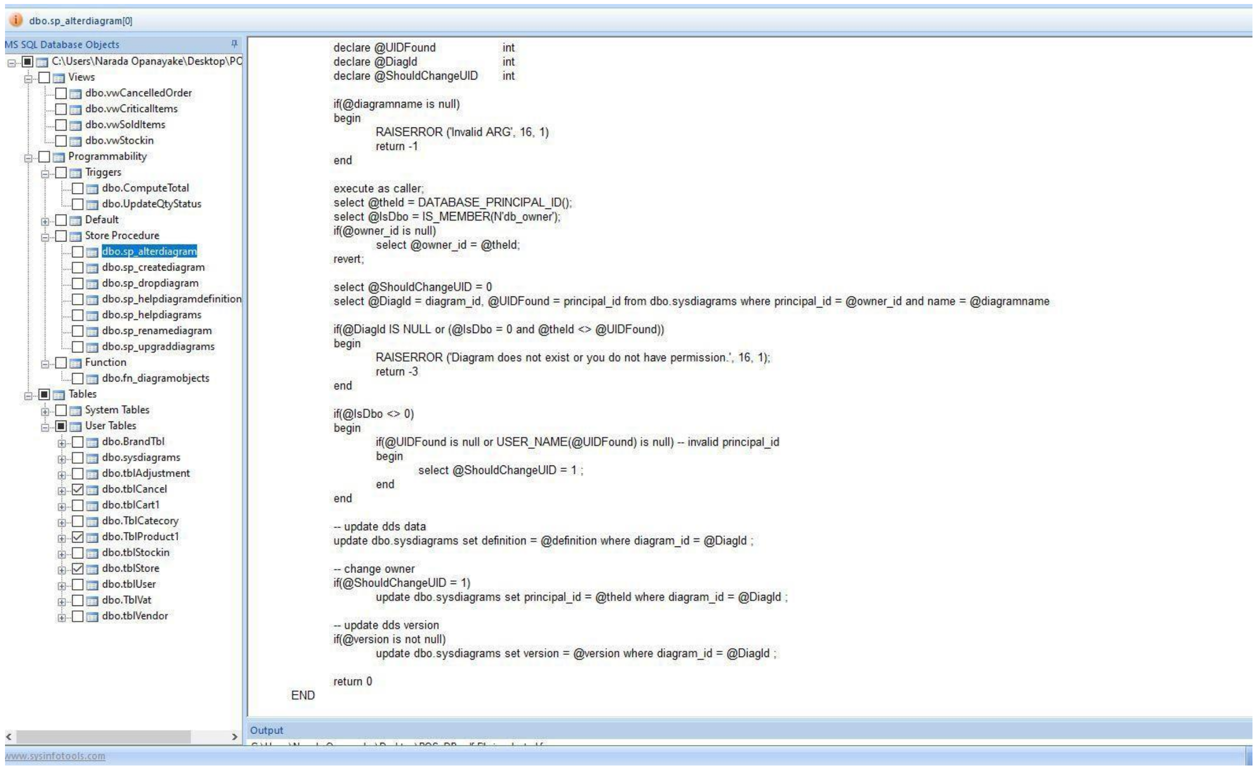
View 4

Create Procedure Statements



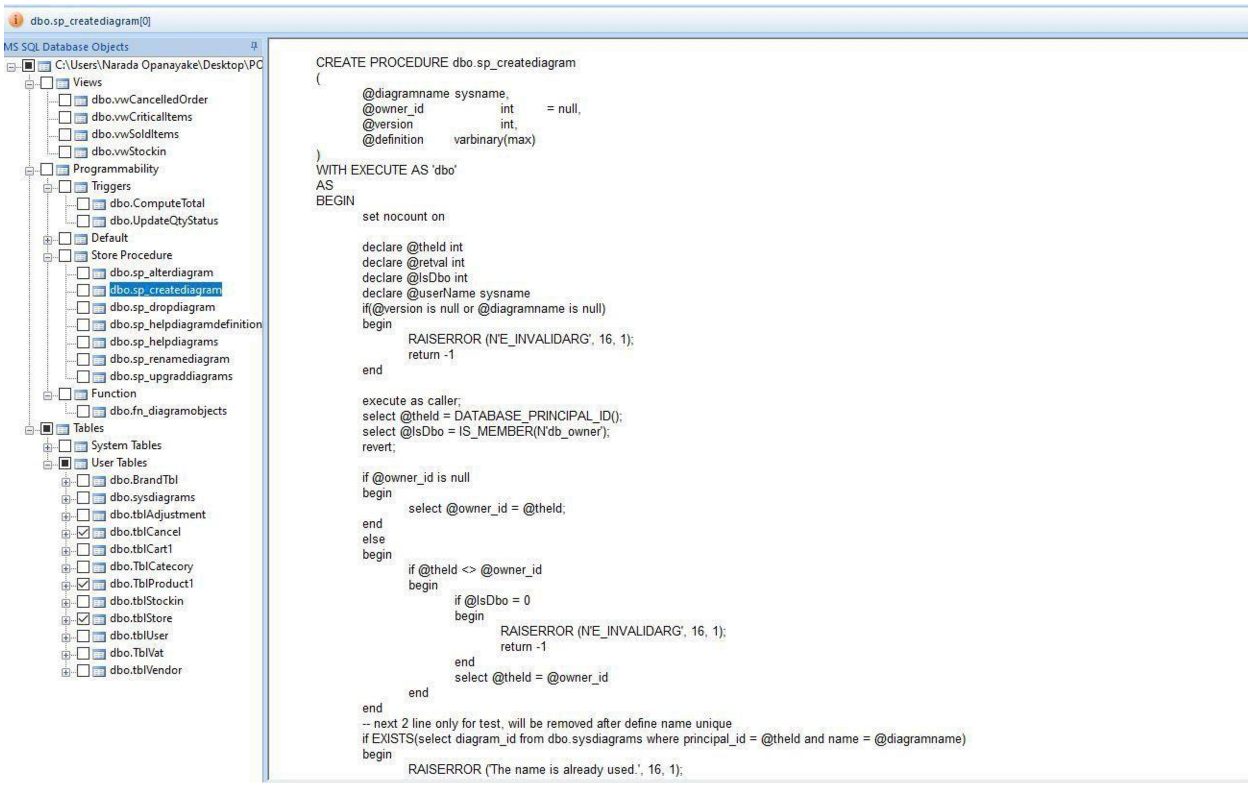
```
CREATE PROCEDURE dbo.sp_alterdiagram
(
        @diagramname sysname,
        @owner_id       int         = null,
        @version        int,
        @definition     varbinary(max)
)
WITH EXECUTE AS 'dbo'
AS
BEGIN
        set nocount on

        declare @theId                  int
        declare @retval         int
        declare @IsDbo                          int

        declare @UIDFound               int
        declare @DiagId                 int
        declare @ShouldChangeUID        int

        if(@diagramname is null)
        begin
                RAISERROR ('Invalid ARG', 16, 1)
                return -1
        end

        execute as caller;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        if(@owner_id is null)
                select @owner_id = @theId;
        revert;

        select @ShouldChangeUID = 0
        select @DiagId = diagram_id, @UIDFound = principal_id from dbo.sysdiagrams where principal_id = @owner_id and name = @diagramname

        if(@DiagId IS NULL or (@IsDbo = 0 and @theId <> @UIDFound))
        begin
                RAISERROR ('Diagram does not exist or you do not have permission.', 16, 1);
                return -3
        end

        if(@IsDbo <> 0)
        begin
                if(@UIDFound is null or USER_NAME(@UIDFound) is null) -- invalid principal_id
                begin
                        select @ShouldChangeUID = 1 ;
```

Procedure1.1



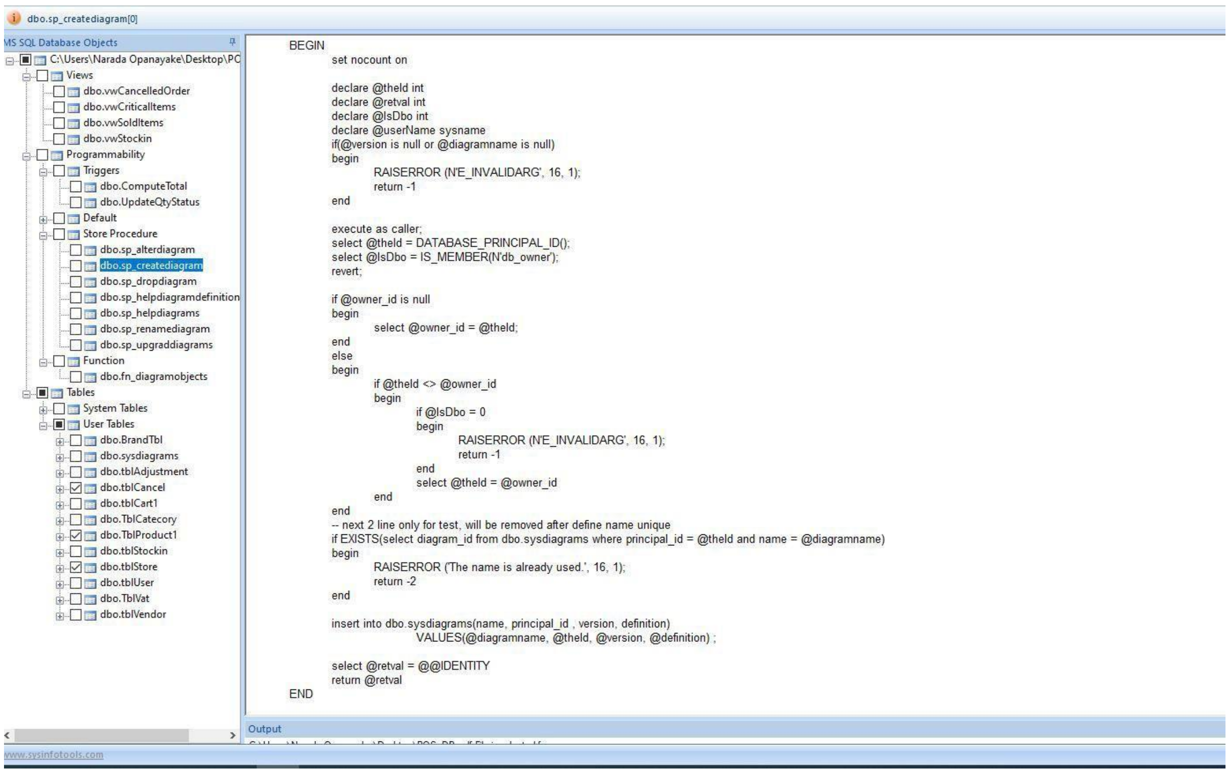
```
        declare @UIDFound               int
        declare @DiagId                 int
        declare @ShouldChangeUID        int

        if(@diagramname is null)
        begin
                RAISERROR ('Invalid ARG', 16, 1)
                return -1
        end

        execute as caller;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        if(@owner_id is null)
                select @owner_id = @theId;
        revert;

        select @ShouldChangeUID = 0
        select @DiagId = diagram_id, @UIDFound = principal_id from dbo.sysdiagrams where principal_id = @owner_id and name = @diagramname

        if(@DiagId IS NULL or (@IsDbo = 0 and @theId <> @UIDFound))
        begin
                RAISERROR ('Diagram does not exist or you do not have permission.', 16, 1);
                return -3
        end

        if(@IsDbo <> 0)
        begin
                if(@UIDFound is null or USER_NAME(@UIDFound) is null) -- invalid principal_id
                begin
                        select @ShouldChangeUID = 1 ;
                end
        end

        -- update dds data
        update dbo.sysdiagrams set definition = @definition where diagram_id = @DiagId ;

        -- change owner
        if(@ShouldChangeUID = 1)
                update dbo.sysdiagrams set principal_id = @theId where diagram_id = @DiagId ;

        -- update dds version
        if(@version is not null)
                update dbo.sysdiagrams set version = @version where diagram_id = @DiagId ;

        return 0
END
```

Procedure1.2

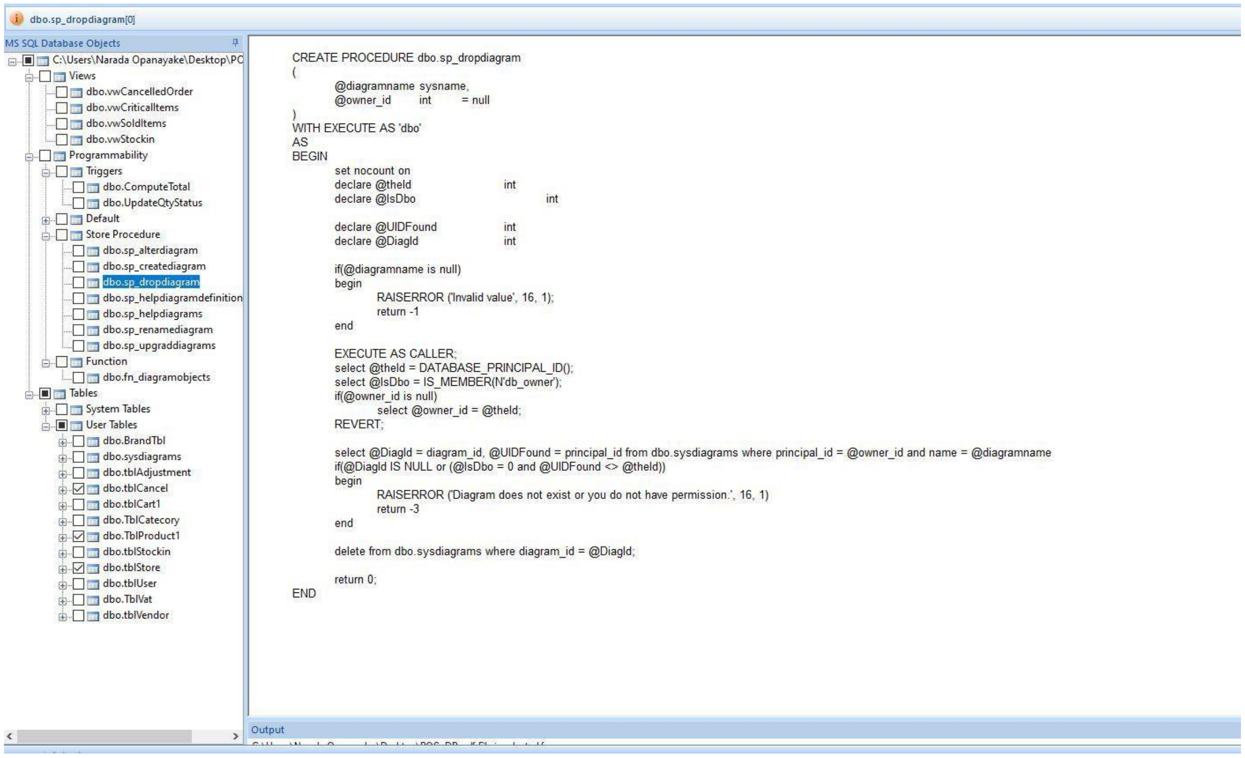


```
CREATE PROCEDURE dbo.sp_creatediagram
(
        @diagramname sysname,
        @owner_id              int        = null,
        @version               int,
        @definition       varbinary(max)
)
WITH EXECUTE AS 'dbo'
AS
BEGIN
        set nocount on

        declare @theId int
        declare @retval int
        declare @IsDbo int
        declare @userName sysname
        if(@version is null or @diagramname is null)
        begin
                RAISERROR (N'E_INVALIDARG', 16, 1);
                return -1
        end

        execute as caller;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        revert;

        if @owner_id is null
        begin
                select @owner_id = @theId;
        end
        else
        begin
                if @theId <> @owner_id
                begin
                        if @IsDbo = 0
                        begin
                                RAISERROR (N'E_INVALIDARG', 16, 1);
                                return -1
                        end
                        select @theId = @owner_id
                end
        end
        -- next 2 line only for test, will be removed after define name unique
        if EXISTS(select diagram_id from dbo.sysdiagrams where principal_id = @theId and name = @diagramname)
        begin
                RAISERROR ('The name is already used.', 16, 1);
```

Procedure2.1

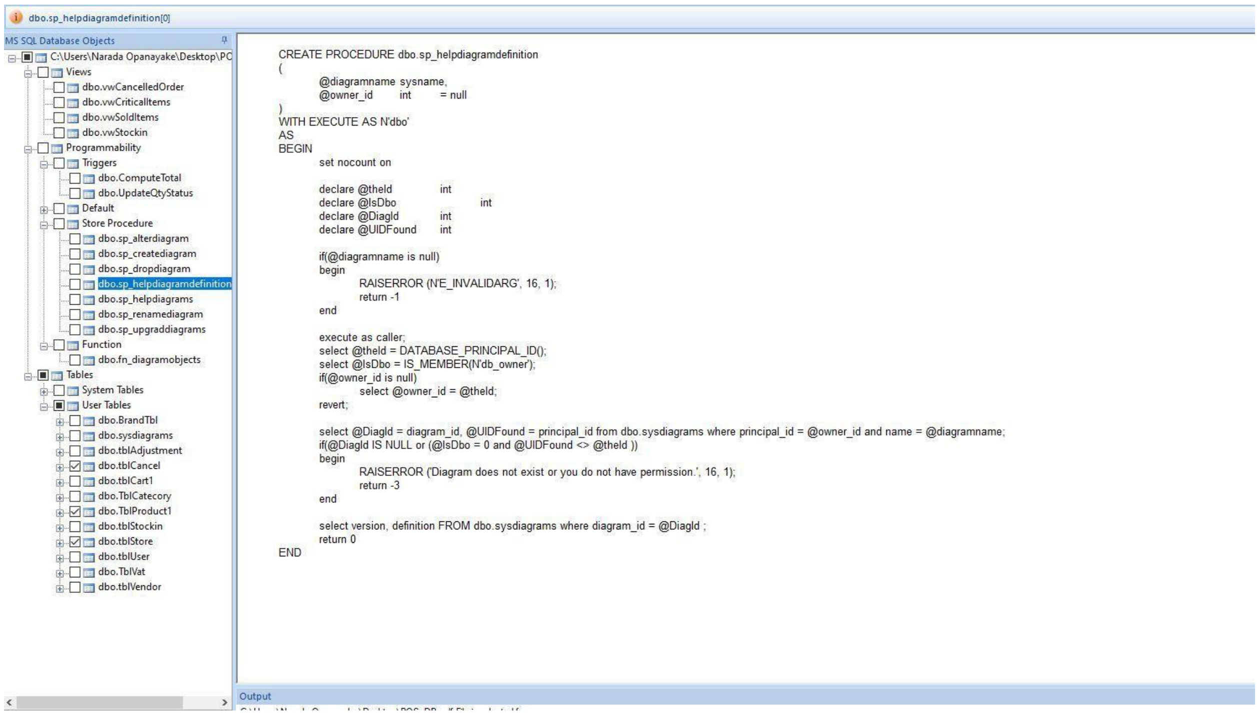


```
BEGIN
        set nocount on

        declare @theId int
        declare @retval int
        declare @IsDbo int
        declare @userName sysname
        if(@version is null or @diagramname is null)
        begin
                RAISERROR (N'E_INVALIDARG', 16, 1);
                return -1
        end

        execute as caller;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        revert;

        if @owner_id is null
        begin
                select @owner_id = @theId;
        end
        else
        begin
                if @theId <> @owner_id
                begin
                        if @IsDbo = 0
                        begin
                                RAISERROR (N'E_INVALIDARG', 16, 1);
                                return -1
                        end
                        select @theId = @owner_id
                end
        end
        -- next 2 line only for test, will be removed after define name unique
        if EXISTS(select diagram_id from dbo.sysdiagrams where principal_id = @theId and name = @diagramname)
        begin
                RAISERROR ('The name is already used.', 16, 1);
                return -2
        end

        insert into dbo.sysdiagrams(name, principal_id , version, definition)
                        VALUES(@diagramname, @theId, @version, @definition) ;

        select @retval = @@IDENTITY
        return @retval
END
```



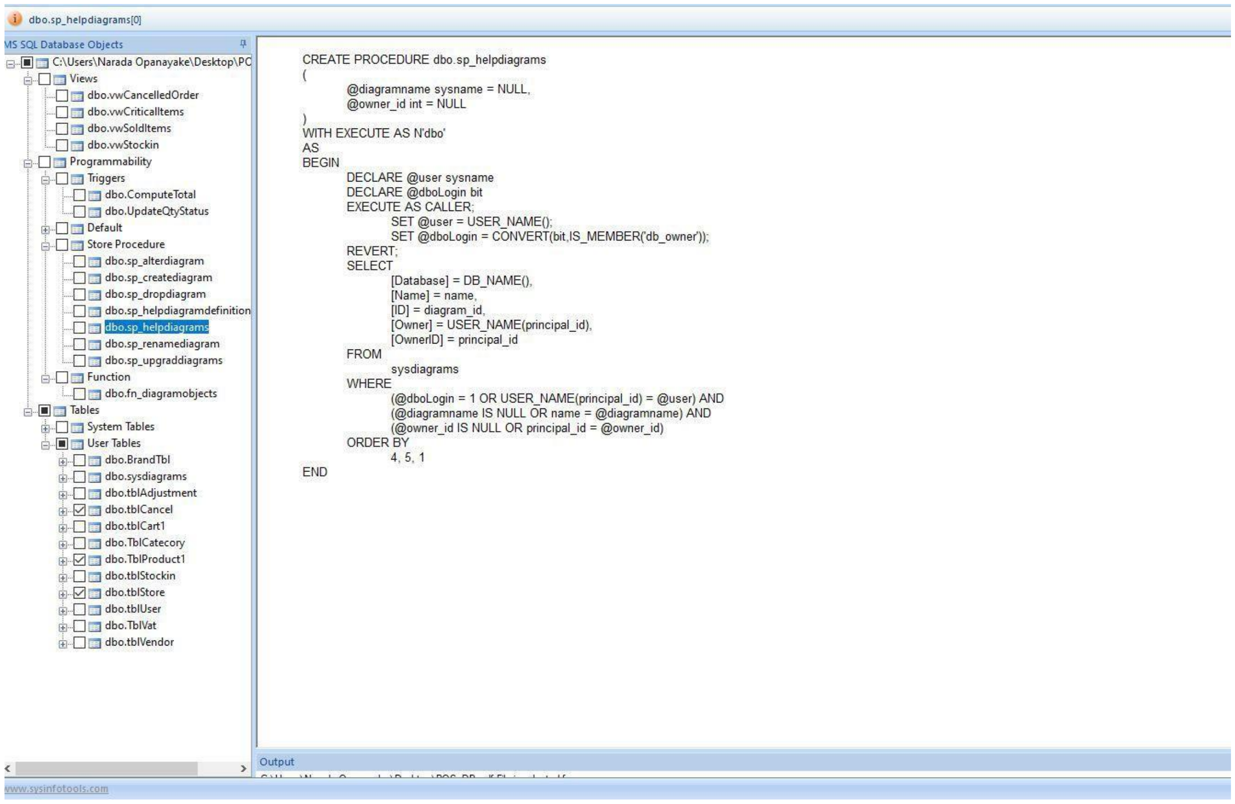
```
CREATE PROCEDURE dbo.sp_dropdiagram
(
        @diagramname sysname,
        @owner_id       int         = null
)
WITH EXECUTE AS 'dbo'
AS
BEGIN
        set nocount on
        declare @theId                  int
        declare @IsDbo                          int

        declare @UIDFound               int
        declare @DiagId                 int

        if(@diagramname is null)
        begin
                RAISERROR ('Invalid value', 16, 1);
                return -1
        end

        EXECUTE AS CALLER;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        if(@owner_id is null)
                select @owner_id = @theId;
        REVERT;

        select @DiagId = diagram_id, @UIDFound = principal_id from dbo.sysdiagrams where principal_id = @owner_id and name = @diagramname
        if(@DiagId IS NULL or (@IsDbo = 0 and @UIDFound <> @theId))
        begin
                RAISERROR ('Diagram does not exist or you do not have permission.', 16, 1)
                return -3
        end

        delete from dbo.sysdiagrams where diagram_id = @DiagId;

        return 0;
END
```

Procedure3



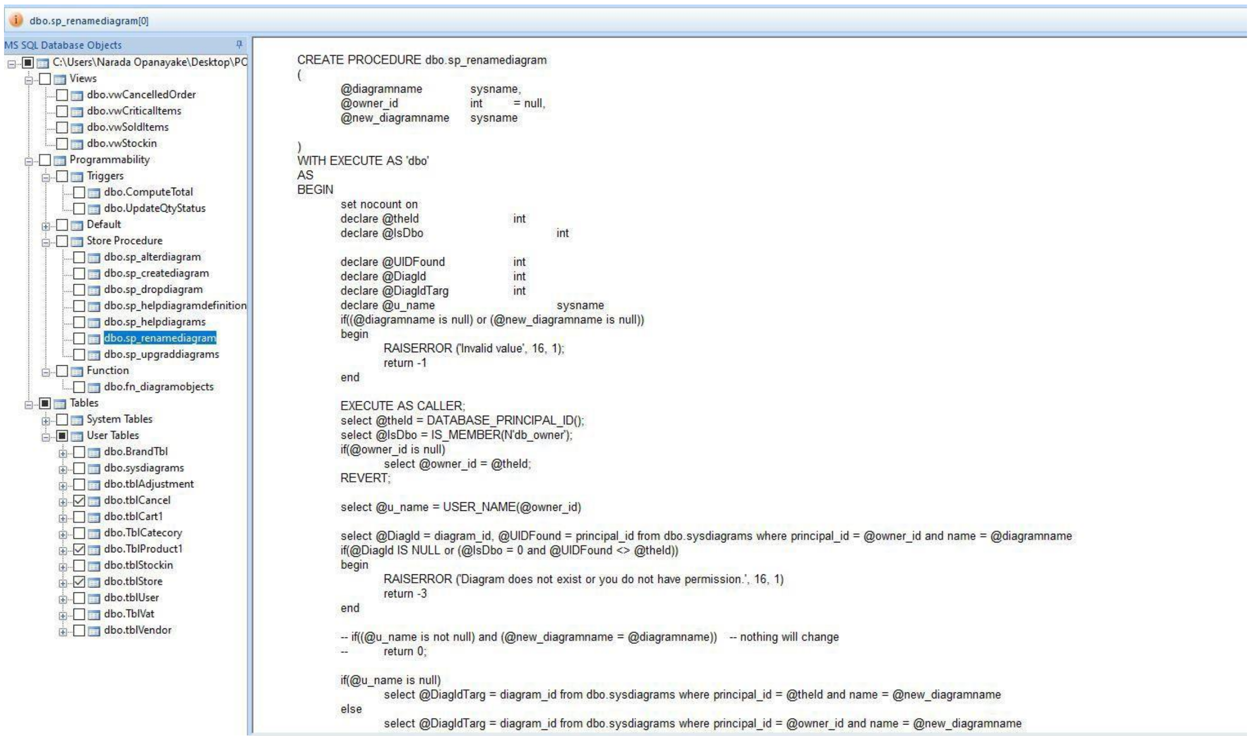
```
CREATE PROCEDURE dbo.sp_helpdiagramdefinition
(
        @diagramname sysname,
        @owner_id       int         = null
)
WITH EXECUTE AS N'dbo'
AS
BEGIN
        set nocount on

        declare @theId          int
        declare @IsDbo                  int
        declare @DiagId         int
        declare @UIDFound       int

        if(@diagramname is null)
        begin
                RAISERROR (N'E_INVALIDARG', 16, 1);
                return -1
        end

        execute as caller;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        if(@owner_id is null)
                select @owner_id = @theId;
        revert;

        select @DiagId = diagram_id, @UIDFound = principal_id from dbo.sysdiagrams where principal_id = @owner_id and name = @diagramname;
        if(@DiagId IS NULL or (@IsDbo = 0 and @UIDFound <> @theId ))
        begin
                RAISERROR ('Diagram does not exist or you do not have permission.', 16, 1);
                return -3
        end

        select version, definition FROM dbo.sysdiagrams where diagram_id = @DiagId ;
        return 0
END
```

Procedure4



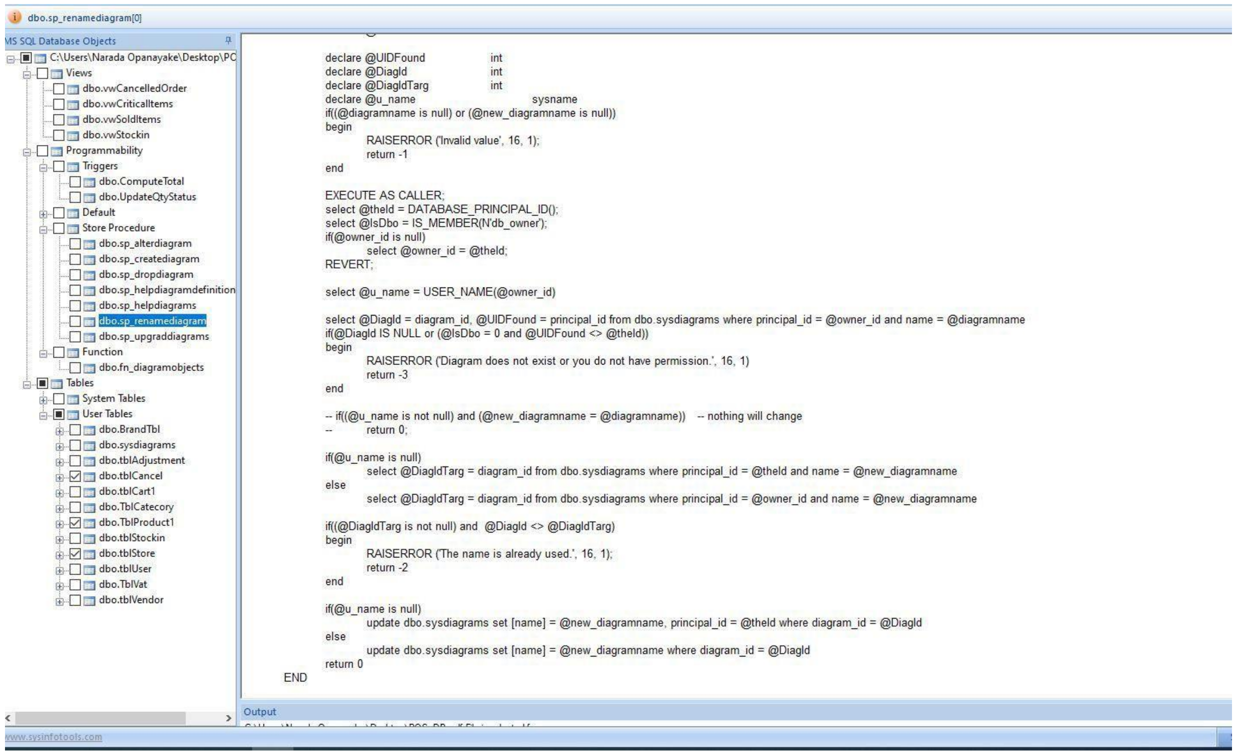
```
CREATE PROCEDURE dbo.sp_helpdiagrams
(
        @diagramname sysname = NULL,
        @owner_id int = NULL
)
WITH EXECUTE AS N'dbo'
AS
BEGIN
        DECLARE @user sysname
        DECLARE @dboLogin bit
        EXECUTE AS CALLER;
                SET @user = USER_NAME();
                SET @dboLogin = CONVERT(bit,IS_MEMBER('db_owner'));
        REVERT;
        SELECT
                [Database] = DB_NAME(),
                [Name] = name,
                [ID] = diagram_id,
                [Owner] = USER_NAME(principal_id),
                [OwnerID] = principal_id
        FROM
                sysdiagrams
        WHERE
                (@dboLogin = 1 OR USER_NAME(principal_id) = @user) AND
                (@diagramname IS NULL OR name = @diagramname) AND
                (@owner_id IS NULL OR principal_id = @owner_id)
        ORDER BY
                4, 5, 1
END
```

Procedure5



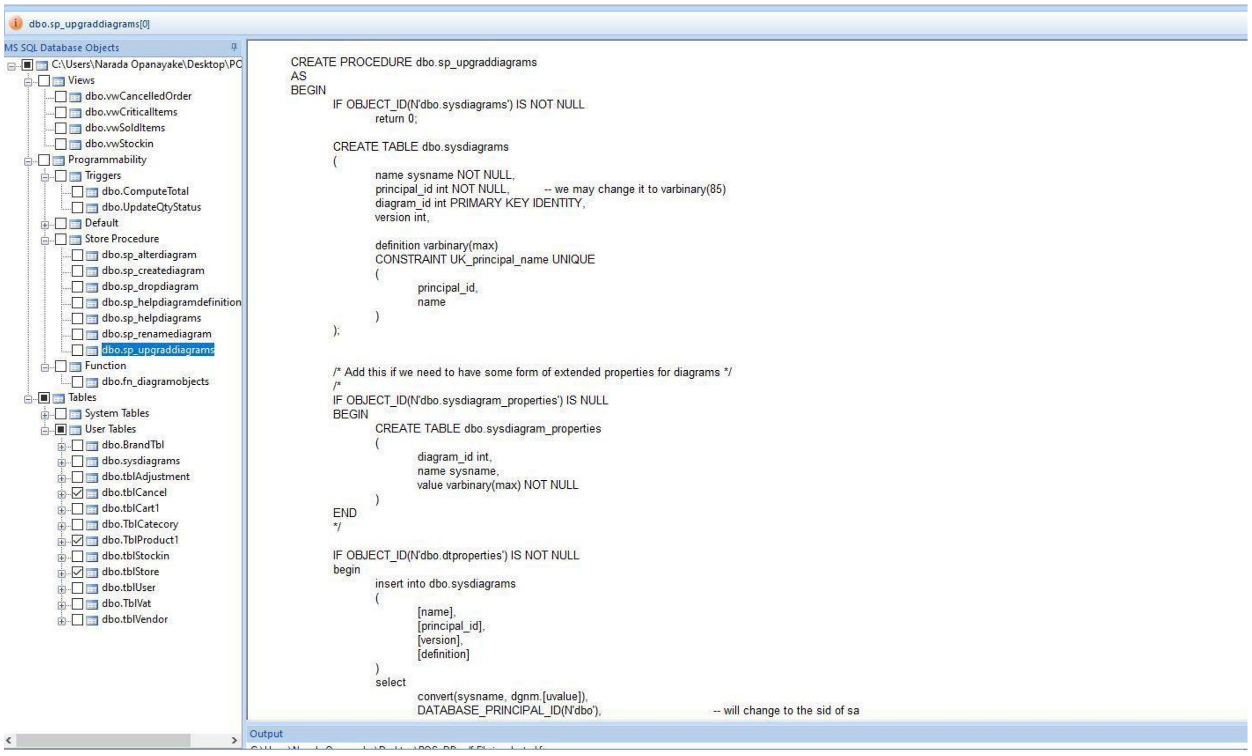
```
CREATE PROCEDURE dbo.sp_renamediagram
(
        @diagramname            sysname,
        @owner_id               int     = null,
        @new_diagramname        sysname

)
WITH EXECUTE AS 'dbo'
AS
BEGIN
        set nocount on
        declare @theId                  int
        declare @IsDbo                          int

        declare @UIDFound               int
        declare @DiagId                 int
        declare @DiagIdTarg             int
        declare @u_name                         sysname
        if((@diagramname is null) or (@new_diagramname is null))
        begin
                RAISERROR ('Invalid value', 16, 1);
                return -1
        end

        EXECUTE AS CALLER;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        if(@owner_id is null)
                select @owner_id = @theId;
        REVERT;

        select @u_name = USER_NAME(@owner_id)

        select @DiagId = diagram_id, @UIDFound = principal_id from dbo.sysdiagrams where principal_id = @owner_id and name = @diagramname
        if(@DiagId IS NULL or (@IsDbo = 0 and @UIDFound <> @theId))
        begin
                RAISERROR ('Diagram does not exist or you do not have permission.', 16, 1)
                return -3
        end

        -- if((@u_name is not null) and (@new_diagramname = @diagramname))     -- nothing will change
        --      return 0;

        if(@u_name is null)
                select @DiagIdTarg = diagram_id from dbo.sysdiagrams where principal_id = @theId and name = @new_diagramname
        else
                select @DiagIdTarg = diagram_id from dbo.sysdiagrams where principal_id = @owner_id and name = @new_diagramname
```

Procedure6.1



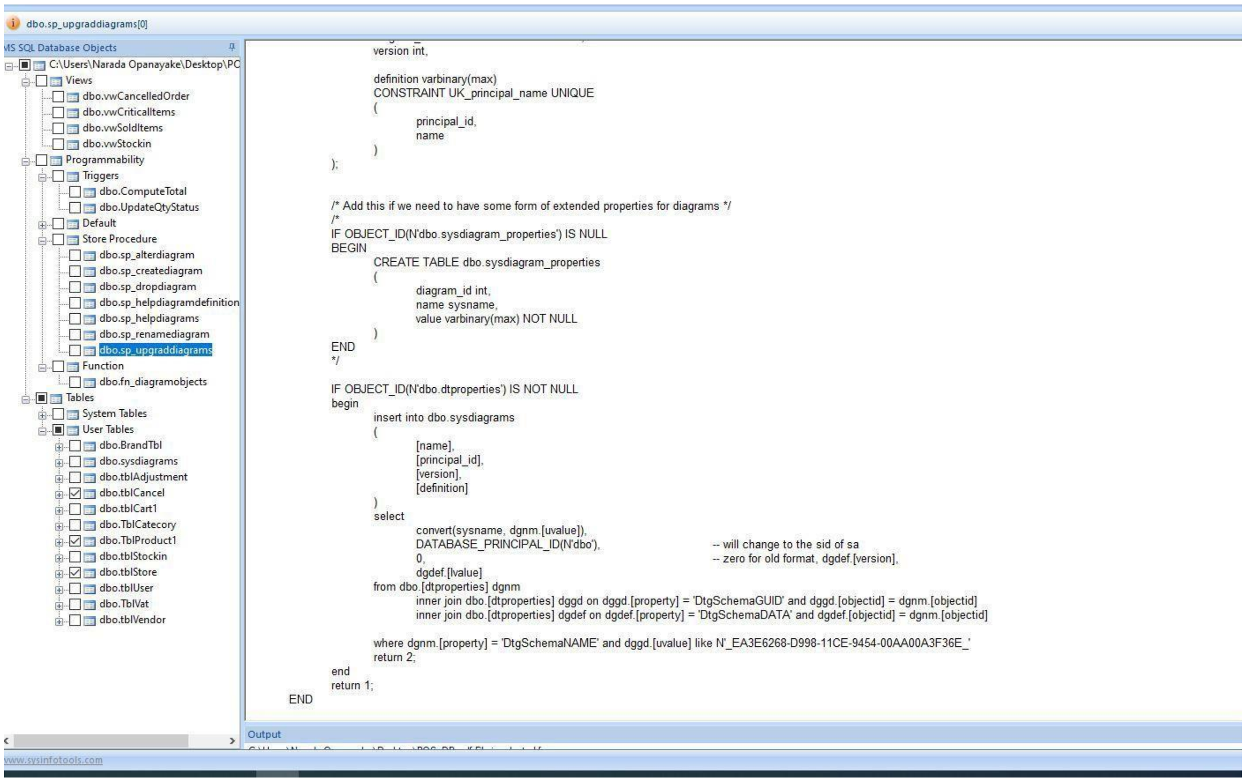
```
        declare @UIDFound           int
        declare @DiagId             int
        declare @DiagIdTarg         int
        declare @u_name                     sysname
        if((@diagramname is null) or (@new_diagramname is null))
        begin
                RAISERROR ('Invalid value', 16, 1);
                return -1
        end

        EXECUTE AS CALLER;
        select @theId = DATABASE_PRINCIPAL_ID();
        select @IsDbo = IS_MEMBER(N'db_owner');
        if(@owner_id is null)
                select @owner_id = @theId;
        REVERT;

        select @u_name = USER_NAME(@owner_id)

        select @DiagId = diagram_id, @UIDFound = principal_id from dbo.sysdiagrams where principal_id = @owner_id and name = @diagramname
        if(@DiagId IS NULL or (@IsDbo = 0 and @UIDFound <> @theId))
        begin
                RAISERROR ('Diagram does not exist or you do not have permission.', 16, 1)
                return -3
        end

        -- if((@u_name is not null) and (@new_diagramname = @diagramname))   -- nothing will change
        --          return 0;

        if(@u_name is null)
                select @DiagIdTarg = diagram_id from dbo.sysdiagrams where principal_id = @theId and name = @new_diagramname
        else
                select @DiagIdTarg = diagram_id from dbo.sysdiagrams where principal_id = @owner_id and name = @new_diagramname

        if((@DiagIdTarg is not null) and  @DiagId <> @DiagIdTarg)
        begin
                RAISERROR ('The name is already used.', 16, 1);
                return -2
        end

        if(@u_name is null)
                update dbo.sysdiagrams set [name] = @new_diagramname, principal_id = @theId where diagram_id = @DiagId
        else
                update dbo.sysdiagrams set [name] = @new_diagramname where diagram_id = @DiagId
        return 0
    END
```

Proceuder6.2



```
    CREATE PROCEDURE dbo.sp_upgraddiagrams
    AS
    BEGIN
        IF OBJECT_ID(N'dbo.sysdiagrams') IS NOT NULL
                return 0;

        CREATE TABLE dbo.sysdiagrams
        (
                name sysname NOT NULL,
                principal_id int NOT NULL,       -- we may change it to varbinary(85)
                diagram_id int PRIMARY KEY IDENTITY,
                version int,

                definition varbinary(max)
                CONSTRAINT UK_principal_name UNIQUE
                (
                        principal_id,
                        name
                )
        );


        /* Add this if we need to have some form of extended properties for diagrams */
        /*
        IF OBJECT_ID(N'dbo.sysdiagram_properties') IS NULL
        BEGIN
                CREATE TABLE dbo.sysdiagram_properties
                (
                        diagram_id int,
                        name sysname,
                        value varbinary(max) NOT NULL
                )
        END
        */

        IF OBJECT_ID(N'dbo.dtproperties') IS NOT NULL
        begin
                insert into dbo.sysdiagrams
                (
                        [name],
                        [principal_id],
                        [version],
                        [definition]
                )
                select
                        convert(sysname, dgnm.[uvalue]),
                        DATABASE_PRINCIPAL_ID(N'dbo'),                  -- will change to the sid of sa
```

Proceuder7.1



```
            version int,

            definition varbinary(max)
            CONSTRAINT UK_principal_name UNIQUE
            (
                principal_id,
                name
            )
        );


        /* Add this if we need to have some form of extended properties for diagrams */
        /*
        IF OBJECT_ID(N'dbo.sysdiagram_properties') IS NULL
        BEGIN
            CREATE TABLE dbo.sysdiagram_properties
            (
                diagram_id int,
                name sysname,
                value varbinary(max) NOT NULL
            )
        END
        */

        IF OBJECT_ID(N'dbo.dtproperties') IS NOT NULL
        begin
            insert into dbo.sysdiagrams
            (
                [name],
                [principal_id],
                [version],
                [definition]
            )
            select
                convert(sysname, dgnm.[uvalue]),
                DATABASE_PRINCIPAL_ID(N'dbo'),                    -- will change to the sid of sa
                0,                                                -- zero for old format, dgdef.[version],
                dgdef.[lvalue]
            from dbo.[dtproperties] dgnm
                inner join dbo.[dtproperties] dggd on dggd.[property] = 'DtgSchemaGUID' and dggd.[objectid] = dgnm.[objectid]
                inner join dbo.[dtproperties] dgdef on dgdef.[property] = 'DtgSchemaDATA' and dgdef.[objectid] = dgnm.[objectid]

            where dgnm.[property] = 'DtgSchemaNAME' and dggd.[uvalue] like N'_EA3E6268-D998-11CE-9454-00AA00A3F36E_'
            return 2;
        end
        return 1;
    END
```

Procedure7.2

Section 4

Critical appraisal of our solution

A critical appraisal of this system would involve evaluating its effectiveness in achieving the goals and objectives of the supermarket. This would include assessing the system's ability to manage inventory, track sales, and generate accurate financial reports. Additionally, the system's ease of use and user-friendliness for both employees and customers would be evaluated. Other factors to consider include the system's scalability, security, and ability to integrate with other systems. The two main benefits of this management system are that it ensures you’re able to fulfill incoming or open orders and raises profits.

- **Saves Money:**
  Understanding stock trends means you see how much of and where you have something in stock so you’re better able to use the stock you have. This also allows you to keep less stock at each location (store, warehouse), as you’re able to pull from anywhere to fulfill orders — all of this decrease costs tied up in inventory and decreases the amount of stock that goes unsold before it’s obsolete.

- **Improves Cash Flow:**
  With proper inventory management, you spend money on inventory that sells, so cash is always moving through the business.

- **Satisfies Customers:**
  One element of developing loyal customers is ensuring they receive the items they want without waiting.

Future implementation

You will find a number of features in our Nano POS system. Basically you can see Admin section and Cashier section. The Cashier section allows the cashier to sell goods, get their daily trading reports, etc. easily. Through the Admin section, the store owner can get reports about the products in his store, identify missing products, add new products and also enter information about the people who provide the products.But we are thinking of updating our system as mentioned below.

1. Introducing a loyalty card system to customers.

This can provide a more attractive and engaging service to the customers. With this card, customers have the opportunity to win more discounts and tags.

2. Method of locating registered and registered customers.

With this, all the information about the registered customers can be obtained through the system. Another convenience here is that if there is any theft in the business, it is easy to find the person concerned. If there is a person who is not registered, his photo will be taken and stored by our system.

3. The possibility of giving a one-time discount for the total amount mentioned in the bill.

Currently, our system has the ability to offer discounts for each product separately, but it is difficult to do this during busy times. It wastes the time of the cashier and the customer. To prevent it. It is our idea to update the system so that a discount can be given for the entire bill at once.