Data Model 1 Exercise

ISM 6218

Due on September 24

The Avengers Team

"We will avenge every problem on our way"

Aitemir Yeskenov (Team Lead)

Nagarjuna Kanneganti

Sai Suraj Argula

Vinay Kumar Reddy Baradi

Table of Contents

Business Process Supported 1

Requirements Described 2

Database diagram 3

Customer And Manager User Stories 4

DDL Operations 5

Constraints 6

DML Customer 7

DML Manager 9

Business Process Supported

For this assignment, we created a database to store information about employees, sales, items being sold, and the receipt info so as to support the transactions after customers purchase items. Every time a customer pays for the order, the stored procedure is being called behind the scenes, which then calculates the total amount for the items as well as taxes and other necessary information.

Requirement Described

Using the store receipt complete the following:

1. Create a user story for the customer.

2. Create a user story for the store manager.

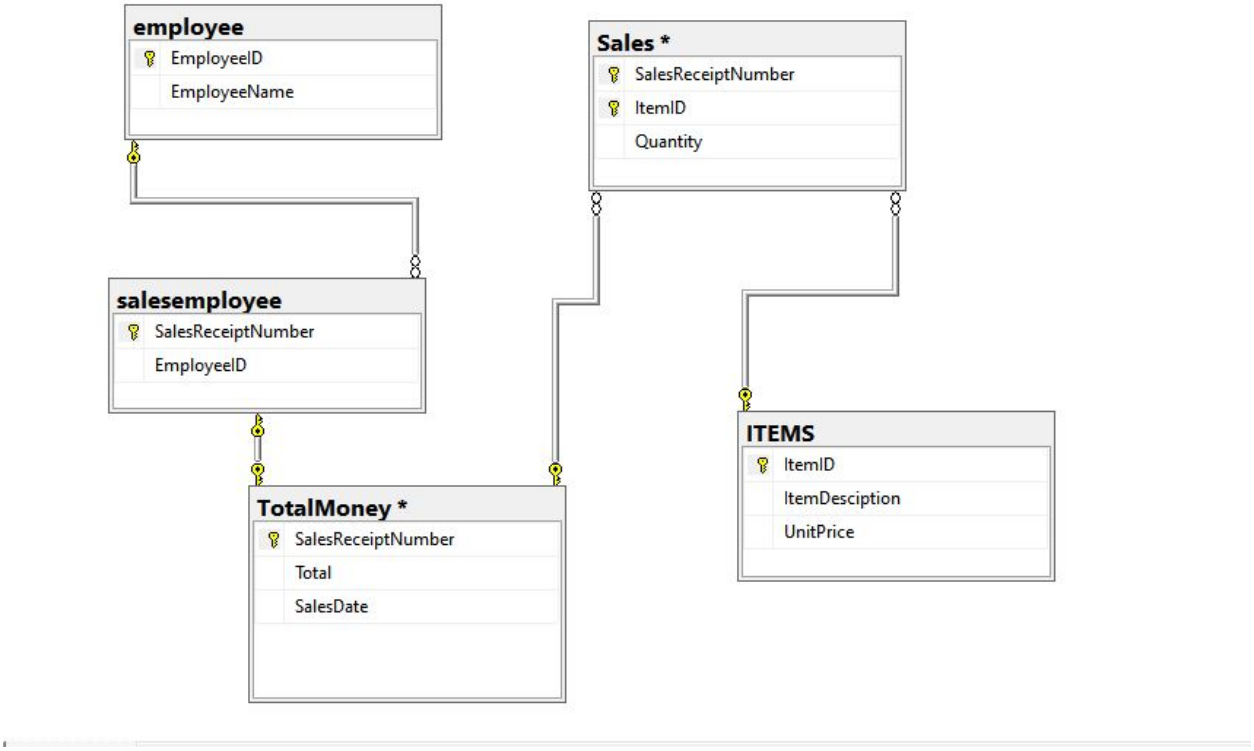
3. DDL design - create tables, columns and constraints supporting both user stories (sales receipt, sales report).

4. DML design - create sprocs that generate the sales information content.

For this assignment you will need to submit the MDF and LDF files, and a DBDD report documenting the user story, derived requirements, user acceptance test queries with logic checks.

Database Diagram

The diagram consists of 5 tables



1) Customer User Story:

The customer pays for the product and wants to receive a receipt with the amount he was charged for. The amount is broken into several parts such as the items of the order, subtotal, tax amount, and the total payment itself. From the customer perspective, they receive nothing but the generated report that shows multiple attributes that eventually comprise the "Payment" that is the sum of all the attributes. This allows the customer to see the details about their purchase and verify the accuracy of the transaction.

2) Manager User Story:

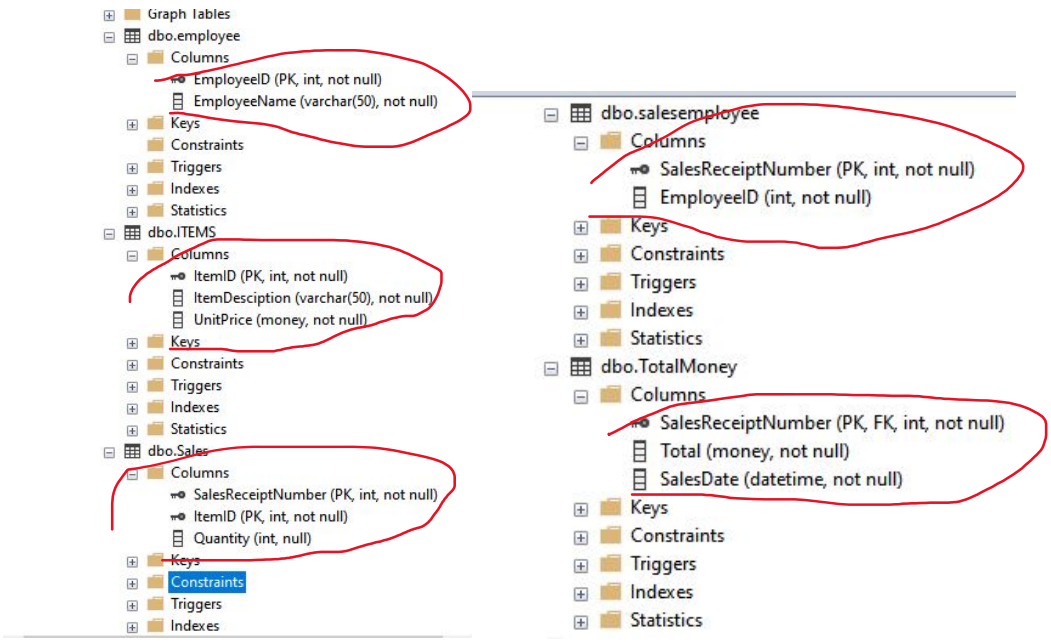
From a manager perspective, things are little bit different. The manager wants to see this "report" / receipt as one of the pieces of data of a bigger dataset containing all transactions per day. In other words, this particular receipt, is one of the records from the number of transactions. The database is being updated constantly as each transaction gets inserted every time a customers pays for their orders.

DDL Operations – Table Creation:

```
CREATE TABLE [dbo].[employee](
	[EmployeeID] [int] NOT NULL,
	[EmployeeName] [varchar](50) NOT NULL,
 CONSTRAINT [PK_employee] PRIMARY KEY NONCLUSTERED
(
	[EmployeeID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[ITEMS](
	[ItemID] [int] NOT NULL,
	[ItemDesciption] [varchar](50) NOT NULL,
	[UnitPrice] [money] NOT NULL,
PRIMARY KEY CLUSTERED
(
	[ItemID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

CREATE TABLE [dbo].[Sales](
	[SalesReceiptNumber] [int] NOT NULL,
	[ItemID] [int] NOT NULL,
	[Quantity] [int] NULL,
 CONSTRAINT [PK_Sales] PRIMARY KEY NONCLUSTERED
(
	[SalesReceiptNumber] ASC,
	[ItemID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[Sales] ADD  DEFAULT ((1)) FOR [Quantity]
GO

ALTER TABLE [dbo].[Sales]  WITH CHECK ADD  CONSTRAINT [FK_ItemID_ItemID] FOREIGN
KEY([ItemID])
REFERENCES [dbo].[ITEMS] ([ItemID])
GO

ALTER TABLE [dbo].[Sales] CHECK CONSTRAINT [FK_ItemID_ItemID]
GO

CREATE TABLE [dbo].[salesemployee](
	[SalesReceiptNumber] [int] NOT NULL,
	[EmployeeID] [int] NOT NULL,
 CONSTRAINT [PK_salesemployee] PRIMARY KEY NONCLUSTERED
(
	[SalesReceiptNumber] ASC
```

```
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[salesemployee]  WITH CHECK ADD  CONSTRAINT [FK_salesemployee_employee]
FOREIGN KEY([EmployeeID])
REFERENCES [dbo].[employee] ([EmployeeID])
GO

ALTER TABLE [dbo].[salesemployee] CHECK CONSTRAINT [FK_salesemployee_employee]
GO

CREATE TABLE [dbo].[TotalMoney](
        [SalesReceiptNumber] [int] NOT NULL,
        [Total] [money] NOT NULL,
        [SalesDate] [datetime] NOT NULL,
 CONSTRAINT [PK_TotalMoney] PRIMARY KEY NONCLUSTERED
(
        [SalesReceiptNumber] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO

ALTER TABLE [dbo].[TotalMoney]  WITH CHECK ADD  CONSTRAINT [FK_TotalMoney_salesemployee]
FOREIGN KEY([SalesReceiptNumber])
REFERENCES [dbo].[salesemployee] ([SalesReceiptNumber])
GO

ALTER TABLE [dbo].[TotalMoney] CHECK CONSTRAINT [FK_TotalMoney_salesemployee]
GO
```

Constraints:



DML Operations – Stored Procedures:

We created stored procedures to present pieces of information:

Customer User Story:

Totalsales presents sum of items in a particular receipt

Grosssales presents total sales multiplied by the percentage of tax (tax value is 1.07)

EmpDetail presents information about the employee that worked on that particular order

(receipt)

```
drop table #receipt
select [dbo].[Sales].[Quantity],[dbo].[ITEMS].[ItemDesciption],[dbo].[ITEMS].[UnitPrice] * [dbo].[Sales].[Quantity] as Itemtot
[dbo].[ITEMS] on [dbo].[Sales].[ItemID] = [dbo].[ITEMS].[ItemID] where [dbo].[Sales].[SalesReceiptNumber]=1;
select * from #receipt

Alter proc totalsales as
begin
declare @totalsales money;
select @totalsales=sum(Itemtotal) from #receipt;
print @totalsales;
end;

Alter proc Grosssales as
begin
declare @Grosssales money;
select @Grosssales=sum(Itemtotal*1.07) from #receipt;
print @Grosssales;
end;

Alter proc EmpDetail as
begin
declare @Empid int;
declare @empname varchar(100);
select @Empid= (select [EmployeeID] from [dbo].[salesemployee] where [dbo].[salesemployee].[SalesReceiptNumber]=1)
select @empname= (select [dbo].[employee].[EmployeeName] from [dbo].[employee] where [dbo].[employee].[EmployeeID]=@Empid)
print 'empid:' + convert(char,@Empid) + 'empname:' + @empname
end;
```

Main stored procedure calls the aforementioned three SPs to print the receipts:

```
Alter PROCEDURE receipt AS
begin
print('               Hotel Vinay International          ')
print('                      Tampa FL                    ')
print('--------------------------------------------------')

exec EmpDetail;
print('--------------------------------------------------')

print('--------------------------------------------------')
exec tablesales

print('Net value :')
exec totalsales;

print('Gross value :')
exec Grosssales;
print('    Thank You! Visit Us again    ')
end;

exec receipt;
```

Output received:

Manager User Story:

We created stored procedures to present pieces of information:

Salesreport stored procedure retrieves all the transactions that have been gone through so far.

```
Alter proc salesreport
as
begin
select [dbo].[TotalMoney].[SalesReceiptNumber],[dbo].[TotalMoney].[Total],[dbo].[TotalMoney].[SalesDate] from [d
print('total revenue')
exec salessofar
print('no of transactions')
exec transactionssofar
end
exec salesreport
```

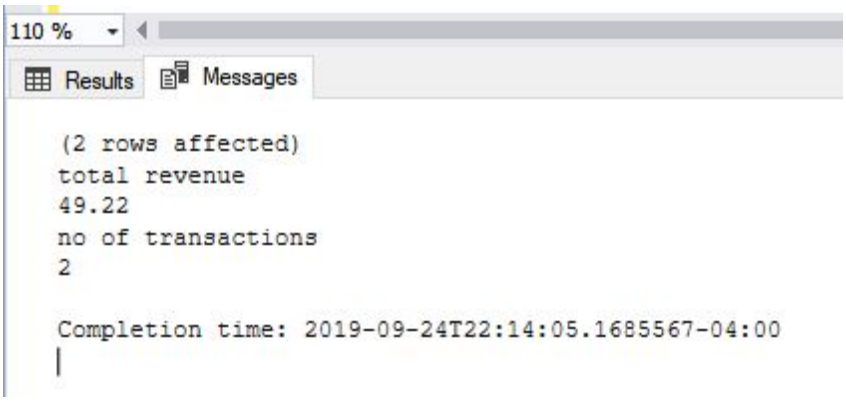
It has nested SPROCS: **salessofar** and **transactionssofar**:

```
--proc for day sales and transaction
Create proc salessofar as
begin
declare @totalsales money;
select @totalsales=sum(Total) from #report;
print @totalsales;
end;
create proc transactionssofar as
begin
declare @transactions int;
select @transactions=count(SalesReceiptNumber) from #report;
print @transactions;
end;
exec transactionssofar
```

The sample output:

Results | Messages

	SalesReceiptNumber	Total	SalesDate
1	1	23.54	2019-09-23 14:14:02.000
2	2	25.68	2019-09-23 14:14:04.000



We are supposed to retrieve 2 results (2 rows affected in messages tab). The logic check for that would be select statement of all records in the table - the total number of records in the table at this point is 2.