

[DATABASE DESIGN AND DEVELOPMENT]

sam
Saw Win Nwe

Contents

Introduction	2
Part 1.....	3
Part 2.....	16
Part 3.....	29
Reference	33

Introduction

This project involves designing a company database with distinct access levels for users and employees. Key tasks include creating an ERD diagram with EDrawMax, normalizing data based on the ERD, and setting up a data validation list. **Company**-specific software is developed with system and hardware requirements outlined for both user and employee use, accompanied by a detailed flowchart illustrating data categories and system flow. Additionally, a recommendation page addresses future improvements, bug fixes, and system enhancements. The database will be created using Microsoft SQL Server 2019, with further details provided in subsequent sections.

Part 1

Create Database

```
Create database db;  
use db;
```

We create a database in Microsoft SQL Server 2019 and name it "Assignment2".

We use "Use assignment" to use the database and insert data in it.

Create Schema

```
use db;  
create schema dbschema;  
---create category table---
```

We create schema and name it "DDDsche" to create a table and insert data in it.

Create category table

```
---create category table---  
create table dbschema.Category(  
  CategoryID char(5) primary key,  
  CategoryName varchar(200) Not Null,)
```

We create the first table and name it "category table" and I create two columns in the table. We name the title "CategoryID(create it as a primary key and char value is 5) and categoryName (not null data type and varchar value (200)". CategoryID is limit and only accept CS including with numbers between 0 to 9.

Insert data in category table

```
drop table dbschema.Category;
---insert into category table---
insert into dbschema.Category values('CTG01','Illume');
insert into dbschema.Category values('CTG02','Histoire');
insert into dbschema.Category values('CTG03','Jeunesse Éternelle');
insert into dbschema.Category values('CTG04','Oracle');
insert into dbschema.Category values('CTG05','Poussière de Fée');
insert into dbschema.Category values('CTG06','Accessoires par Charade');
insert into dbschema.Category values('CTG07','Mode par Vertu');
insert into dbschema.Category values('CTG08','Couture par Étincelle');
insert into dbschema.Category values('CTG09','Couture par Parangon');
insert into dbschema.Category values('CTG10','Mode par Impératrice');
insert into dbschema.Category values('CTG11','Virtue');
insert into dbschema.Category values('CTG12','Paradox');
insert into dbschema.Category values('CTG13','Halcyon');
insert into dbschema.Category values('CTG14','Temptation');
insert into dbschema.Category values('CTG15','Jumpsync');
```

we insert data in category table which is base on “DDDsche” with limits words and value as we create to categoryID and categoryName. There are 2 coloum and first column is customerID column and the ID start with CS including with numbers and total 5 words and there are total 15 customers. Second column is CustomerName and there are a lot of customer name which used as “String” data type “Varchar” data type which allows 200 words of Customer’s name.

Check the category table

```
select * from DDDsche.Category;
```

We check the list but writing this code in Microsoft SQL server 2019 and check and table and results are shown below.

Results		Messages	
	CategoryID	CategoryName	
1	CTG01	Illume	
2	CTG02	Histoire	
3	CTG03	Jeunesse Éternelle	
4	CTG04	Oracle	
5	CTG05	Poussière de Fée	
6	CTG06	Accessoires par Charade	
7	CTG07	Mode par Vertu	
8	CTG08	Couture par Étincelle	
9	CTG09	Couture par Parangon	
10	CTG10	Mode par Impératrice	
11	CTG11	Virtue	

Create Item table

```

---create table dbsche.Item---
create table dbschema.Item(
ItemID char(4) primary key,
ItemName varchar(100) Not Null,
ItemUnitPrice money Not Null,
CategoryID char(5) foreign key references dbschema.Category(CategoryID));
---insert into dbsche.Item---

```

We create another table base on same schema "DDDsche" and we name the table "Item" as item table for the category table above which is related with customer and the item customer buy. There are 4 coloum in this table "ItemID" (which is primary key and same data type as CustomerID which is with char 5 values), ItemName(which is "varchar" data type with (100) words allowance no null), ItemUnitPrice(with "money" data type and only both user and employee can write money data, not null) and following up with CategoryID from category table from above but in this table, it is use as foreign key and relay data from category table.

Insert item table

```

insert into dbschema.Item values('ID10','Jeans',30000,'CTG01'),('ID22','Shirts and pants',70000,'CTG02'),('ID50','Underwaer',10000,'CTG09'),
('ID69','Jumper Suits',40000,'CTG13'),('ID13','Woman Suits',150000,'CTG03'),('ID52','Jackets',90000,'CTG11'),('ID53','Bra',20000,'CTG08'),
('ID59','Jumper',21000,'CTG12'),('ID01','Belts and Ties',20000,'CTG04'),('ID55','Body Suits',170000,'CTG05'),('ID54','Cardigan',60000,'CTG10'),
('ID58','Bloomer',45000,'CTG07'),('ID57','Coat',65000,'CTG06'),('ID56','Play Suits',40000,'CTG14');

```

Result of item table

Results		Messages		
	ItemID	ItemName	ItemUnitPrice	CategoryID
1	ID01	Belts and Ties	20000.00	CTG04
2	ID10	Jeans	30000.00	CTG01
3	ID13	Woman Suits	150000.00	CTG03
4	ID22	Shirts and pants	70000.00	CTG02
5	ID50	Underwaer	10000.00	CTG09
6	ID51	Tshirt	20000.00	CTG15
7	ID52	Jackets	90000.00	CTG11
8	ID53	Bra	20000.00	CTG08
9	ID54	Cardigan	60000.00	CTG10
10	ID55	Body Suits	170000.00	CTG05
11	ID56	Play Suits	40000.00	CTG14

There will be itemID column fills with itemID from company, ItemName with every type of product in the company, ItemUnitPrice with different item prices for different items and categoryID from category table.

Staff table

Create staff table

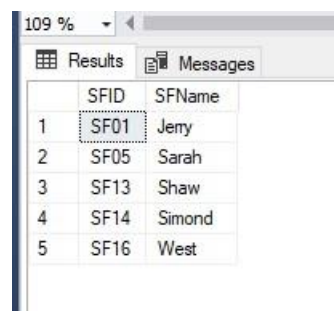
```
---create table dbschema.SF---  
create table dbschema.SF(  
  SFID char(4) primary key,  
  SFName varchar(100))
```

Create staff table within "DDDsche" shema and create table by using the code above and name the table as "DDDsche.SF" which contain 2 columns with "SFID(with "Char" data type and word limitation is "4" for every data in that column and exits as primary key in this table) and SFName(with "varchar" data type and (100) words allowance) .

Insert staff table

```
---insert into dbschema.SF---  
insert into dbschema.SF values('SF01','Jerry');  
insert into dbschema.SF values('SF05','Sarah'),  
  ('SF13','Shaw'),('SF14','Simond'),  
  ('SF16','West');
```

Result of staff table



The screenshot shows a database query result window with a zoom level of 109%. It displays a table with two columns: SFID and SFName. The table contains five rows of data. The first row is highlighted with a mouse cursor.

	SFID	SFName
1	SF01	Jerry
2	SF05	Sarah
3	SF13	Shaw
4	SF14	Simond
5	SF16	West

From above table, there are 2 columns and the one with SFID column has all the data ID from company with the data type and this is also known as

employeeID from the company. The other one is SFName column and all the employee name from company can be seen here but their name and their ID are connected and this can't be changed.

Customer table Create customer table

```

---create table dbschema.CUStomer---
create table dbschema.CUStomer(
  CUStomerID char(5) primary key,
  CUStomerName varchar(100) Not Null)

```

We create the customer table using code from above within "DDDsche" schema and name the table as "DDDsche.Cus". this table contain 2 columns. CustomerID column (with "char" data type with 5 words limitation and exits as primary key in the table) and CustomerName(with "varchar" data type with (100) words allowance , not null).

Insert customer table

```

---insert into dbschema.CUStomer---
insert into dbschema.CUStomer values('CUS01','Selena');
insert into dbschema.CUStomer values('CUS04','Maya'),('CUS06','Billie'),('CUS07','Taylor'),
('CUS11','Polly'),('CUS12','Corbyn'),('CUS13','Don'),('CUS14','Thelma '),('CUS15','Wheatly');

select * from dbschema.CUStomer;

```

Result of customer table

Results		Messages
	CUSomerID	CUSomerName
1	CUS01	Selena
2	CUS04	Maya
3	CUS06	Billie
4	CUS07	Taylor
5	CUS11	Polly
6	CUS12	Corbyn
7	CUS13	Don
8	CUS14	Thelma
9	CUS15	Wheatly

From result table above, there are 2 columns and the first column is "CustomerID" and we can see or put all the customerID from the company and the second column is "CustomerName" which is relate to the "CustomerID" column (primary key) and it cannot be changed due or duplicate same ID with different name because the ID words 1 ID per person and we can check all the customer name with their ID from the company.

Order table Create order table

```

---create table dbschema.Ord---
create table dbschema.Ord(
  OrderNumber char(4) primary key,
  OrderDate Date Not Null,
  DeliveryDate Date Not Null,
  DeliveryFee smallmoney Not Null,
  Discount int NOT NULL,
  DeliveryTime Time NOT NULL,
  Tax FLOAT NOT NULL)
Alter table dbschema.Ord add SFID char(4)
Alter table dbschema.Ord add constraint FK_SF_SFID foreign key (SFID)
references dbschema.SF(SFID);
Alter table dbschema.Ord add CustomerID char(5)
Alter table dbschema.Ord add constraint FK_CUS_CustomerID foreign key (CustomerID)
references dbschema.Customer(CustomerID);

```

We create the order table using code from above with "DDDsche" schema and name the table "DDDsche.Ord". There are 9 columns in totoal in this table. The first column also know as primary key is OrderNumber column (with 'char' data type (4) words allowance), following up with OrderDate column(with "Date" data type and user or employee can only type date for this column, not null), Delivery Date column(with "Date" data type same as orderdate column and this column can also type date, not null), DeliveryFee column(with "money" data type and in this column user or employee can only type money same as ItemUnitPrice from Item table but this money data type is low because it is delivery fee for the items to delivery for the customers), not null, Discount column(with "int" data type, simple any numbers data can be exits in this column,, not null), DeliveryTime column (with "Time" data type which is user or employee can enter only time in this column, not null) and Tax column(with "FLOAT" data type, only float numbers are allow because this is tax money for customer and the higher the price of the item that customer

buys, the higher percentage of the Tax money and more float can be extis). This table is also connect to the staff table with SFID as foreign key and CustomerID as foreign key from customer table and insert data as below.

Insert order table

```
insert into dbschema.Ord values('OD06','2022/9/20','2022/12/21',2000,0,'11:00',153.580,'SF01','CUS06');
insert into dbschema.Ord values('OD07','2022/9/20','2022/10/1',10000,0,'11:00',153.580,'SF05','CUS07');
insert into dbschema.Ord values('OD21','2022/10/3','2022/10/5',4000,0,'10:10',153.580,'SF14','CUS01');
insert into dbschema.Ord values('OD11','2022/10/4','2022/10/2',2000,0,'9:10',153.580,'SF13','CUS11');
insert into dbschema.Ord values('OD12','2022/10/10','2022/10/2',4000,0,'10:45',153.580,'SF13','CUS12');
insert into dbschema.Ord values('OD13','2022/10/13','2022/10/15',3000,0,'10:30',153.580,'SF13','CUS13');
insert into dbschema.Ord values('OD24','2022/10/14','2022/10/18',10000,0,'10:00',153.580,'SF16','CUS04');
insert into dbschema.Ord values('OD14','2022/10/15','2022/10/17',4000,0,'12:50',153.580,'SF13','CUS14');
insert into dbschema.Ord values('OD15','2022/10/18','2022/10/20',2000,0,'11:36',153.580,'SF13','CUS15');
insert into dbschema.Ord values('OD26','2022/10/21','2022/10/23',2000,0,'1:00',153.580,'SF16','CUS06');
insert into dbschema.Ord values('OD16','2022/10/22','2022/10/26',3000,0,'10:15',153.580,'SF13','CUS06');
insert into dbschema.Ord values('OD27','2022/11/2','2022/11/5',3000,0,'2:45',153.580,'SF16','CUS07');
```

Result of order table

Results		Messages								
	OrderNumber	OrderDate	DeliveryDate	DeliveryFee	Discount	DeliveryTime	Tax	SFID	CustomerID	
1	OD04	2022-09-20	2022-12-21	2000.00	0	10:00:00.0000000	153.58	SF01	CUS04	
2	OD06	2022-09-20	2022-12-21	2000.00	0	11:00:00.0000000	153.58	SF01	CUS06	
3	OD07	2022-09-20	2022-10-01	10000.00	0	11:00:00.0000000	153.58	SF05	CUS07	
4	OD11	2022-10-04	2022-10-02	2000.00	0	09:10:00.0000000	153.58	SF13	CUS11	
5	OD12	2022-10-10	2022-10-02	4000.00	0	10:45:00.0000000	153.58	SF13	CUS12	
6	OD13	2022-10-13	2022-10-15	3000.00	0	10:30:00.0000000	153.58	SF13	CUS13	
7	OD14	2022-10-15	2022-10-17	4000.00	0	12:50:00.0000000	153.58	SF13	CUS14	
8	OD15	2022-10-18	2022-10-20	2000.00	0	11:36:00.0000000	153.58	SF13	CUS15	
9	OD16	2022-10-22	2022-10-26	3000.00	0	10:15:00.0000000	153.58	SF13	CUS06	
10	OD21	2022-10-03	2022-10-05	4000.00	0	10:10:00.0000000	153.58	SF14	CUS01	
11	OD24	2022-10-14	2022-10-18	10000.00	0	10:00:00.0000000	153.58	SF16	CUS04	

Query executed successfully. DESKTOP-A1U

In this table, there 9 columns in total and the first column as primary which is connect to customer ID and order number is base on order date which is date between September to December and it connected with OederID, CustomerID, SFID and Order Date and Staff will calculate the day for delivery and add delivery date data and delivery date is also dates between September to December and if customer is whole buyer, the staff will put discount and base

on the price, they will also calculate tax and as foreign key SFID is in this table too. **Order In voice table**

Create order in voice

```

---create table OrderInvoice---
create table dbschema.OrderInvoice(
  Ordernumber char(4) foreign key references dbschema.Ord(Ordernumber),
  ItemID char(4) foreign key references dbschema.Item(ItemID),
  Quantity int NOT NULL);
drop table dbschema.OrderInvoice
---insert into OrderInvoice---

```

We create order in voice table with "DDDsche" schema and name the table "DDDsche.OrdInvoice" with 3 columns. This table is related to the order table and item table and no primary key. The first column is Order number from order table and it exists as foreign key in this table and ItemID from item table as foreign key and last one is quantity of the item.

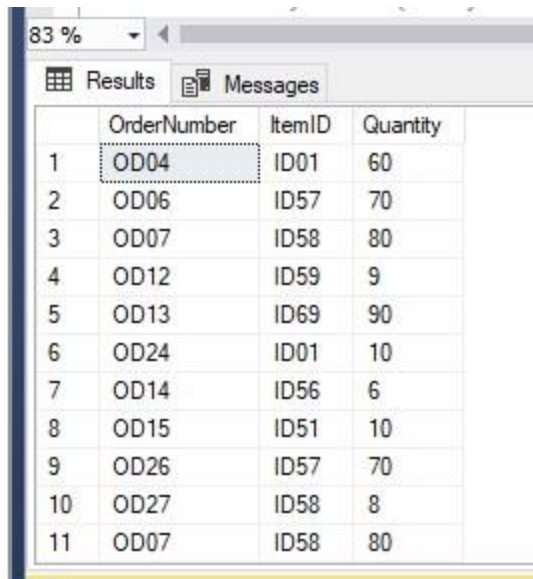
Insert order in voice table

```

drop table dbschema.OrderInvoice
---insert into OrderInvoice---
insert into dbschema.OrderInvoice values('OD04','ID01',60);
insert into dbschema.OrderInvoice values('OD06','ID57',70);
insert into dbschema.OrderInvoice values('OD07','ID58',80);
insert into dbschema.OrderInvoice values('OD21','ID10',1);
insert into dbschema.OrderInvoice values('OD11','ID52',2);
insert into dbschema.OrderInvoice values('OD12','ID59',9);
insert into dbschema.OrderInvoice values('OD13','ID69',90);
insert into dbschema.OrderInvoice values('OD24','ID01',10);
insert into dbschema.OrderInvoice values('OD14','ID56',6);
insert into dbschema.OrderInvoice values('OD15','ID51',10);
insert into dbschema.OrderInvoice values('OD26','ID57',70);
insert into dbschema.OrderInvoice values('OD16','ID57',7);
insert into dbschema.OrderInvoice values('OD27','ID58',8);

```

Result of order in voice table



	OrderNumber	ItemID	Quantity
1	OD04	ID01	60
2	OD06	ID57	70
3	OD07	ID58	80
4	OD12	ID59	9
5	OD13	ID69	90
6	OD24	ID01	10
7	OD14	ID56	6
8	OD15	ID51	10
9	OD26	ID57	70
10	OD27	ID58	8
11	OD07	ID58	80

In this table, there is order number as foreign key connect to ItemID from customer data and quantity.

Query 1

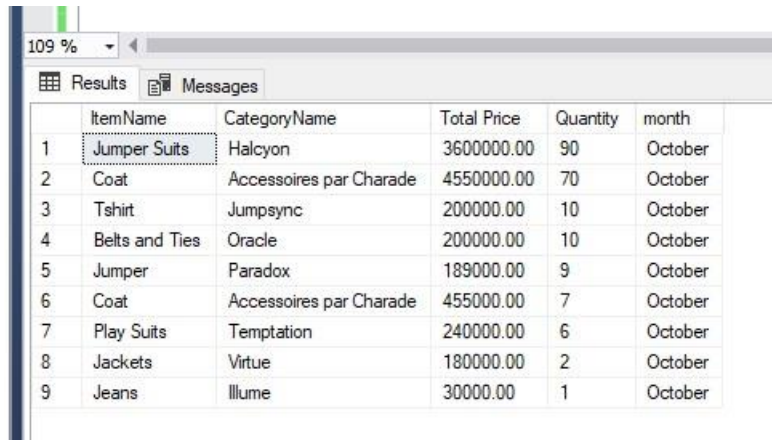
```

----Query1---
select top 15 dbschema.Item.ItemName,dbschema.Category.CategoryName,sum(dbschema.Item.ItemUnitPrice* dbschema.OrderInvoice.Quantity)
as "Total Price",sum(dbschema.OrderInvoice.Quantity) as "Quantity", datename(month,dbschema.Ord.OrderDate) as month
from dbschema.Item inner join dbschema.Category on dbschema.Item.CategoryID = dbschema.Category.CategoryID inner join dbschema.OrderInvoice
on dbschema.OrderInvoice.ItemID = dbschema.Item.ItemID inner join dbschema.Ord on dbschema.OrderInvoice.OrderNumber = dbschema.Ord.OrderNumber
where month(dbschema.Ord.OrderDate) = month(getdate())
group by dbschema.Item.ItemID,dbschema.Item.ItemName,dbschema.Category.CategoryName,dbschema.Ord.OrderDate
order by sum(dbschema.OrderInvoice.Quantity) DESC

```

In this query 1, we collect Item and ItemID from item table and quantity from order in voice table and sum those two for total quantity number and name the result as "Total Ordering quantity". We collect Order in voice table data and sum the ItemUnitPrice from item table with Quantity from order in voice table and name it "Total prices." And we put month only from order table but we take month data only to show which item was order in which months.

Result of the query



	ItemName	CategoryName	Total Price	Quantity	month
1	Jumper Suits	Halcyon	3600000.00	90	October
2	Coat	Accessoires par Charade	4550000.00	70	October
3	Tshirt	Jumpsync	200000.00	10	October
4	Belts and Ties	Oracle	200000.00	10	October
5	Jumper	Paradox	189000.00	9	October
6	Coat	Accessoires par Charade	455000.00	7	October
7	Play Suits	Temptation	240000.00	6	October
8	Jackets	Virtue	180000.00	2	October
9	Jeans	Illume	30000.00	1	October

Query 2

```

---Query2---
select distinct(dbschema.Item.ItemID),dbschema.Item.ItemName,dbschema.Category.CategoryName,sum(dbschema.OrderInvoice.Quantity * dbschema.Item.ItemUnitPrice)
as "Total Price",sum(dbschema.OrderInvoice.Quantity) as "Total ordering amount",datetime(month,dbschema.Ord.OrderDate) as Month,year(dbschema.Ord.OrderDate)as Year
from dbschema.Item inner join dbschema.Category on dbschema.Item.CategoryID = dbschema.Category.CategoryID inner join dbschema.OrderInvoice
on dbschema.OrderInvoice.ItemID = dbschema.Item.ItemID inner join dbschema.Ord on dbschema.OrderInvoice.OrderNumber=dbschema.Ord.OrderNumber
where year(dbschema.Ord.OrderDate) = 2022
group by dbschema.Item.ItemID,dbschema.Item.ItemName,dbschema.Category.CategoryName,dbschema.Ord.OrderDate

```

In this query 2, we select the itemID from item table and CategoryName from category table and sum it with quantity from order in voice table with itemUnitPrice from item table and name it as "Total Price" and from the table above we have "Total Ordering Quantity" and from order table. We use month and year data and separated in month and year. The result will show below.

Result of the query

	ItemID	ItemName	CategoryName	Total Price	Total ordering amount	Month	Year
1	ID01	Belts and Ties	Oracle	200000.00	10	October	2022
2	ID01	Belts and Ties	Oracle	1200000.00	60	September	2022
3	ID10	Jeans	Illume	30000.00	1	October	2022
4	ID51	Tshirt	Jumpsync	200000.00	10	October	2022
5	ID52	Jackets	Virtue	180000.00	2	October	2022
6	ID56	Play Suits	Temptation	240000.00	6	October	2022
7	ID57	Coat	Accessoires par Charade	455000.00	7	October	2022
8	ID57	Coat	Accessoires par Charade	4550000.00	70	October	2022
9	ID57	Coat	Accessoires par Charade	4550000.00	70	September	2022
10	ID58	Bloomer	Mode par Vertu	360000.00	8	November	2022
11	ID58	Bloomer	Mode par Vertu	3600000.00	80	September	2022

Query 3

```
-- Query 3 ----
select top 15 dbschema.Item.ItemName,dbschema.Category.CategoryName,sum(dbschema.Item.ItemUnitPrice*dbschema.OrderInvoice.Quantity)
as "Total Price",sum(dbschema.OrderInvoice.Quantity) as "Quantity", datename(month,dbschema.Ord.OrderDate) as month
from dbschema.Item inner join dbschema.Category on dbschema.Item.CategoryID = dbschema.Category.CategoryID inner join dbschema.OrderInvoice
on dbschema.OrderInvoice.ItemID = dbschema.Item.ItemID inner join dbschema.Ord on dbschema.OrderInvoice.OrderNumber = dbschema.Ord.OrderNumber
where month(dbschema.Ord.OrderDate) = month(getdate())
group by dbschema.Item.ItemID,dbschema.Item.ItemName,dbschema.Category.CategoryName,dbschema.Ord.OrderDate
order by sum(dbschema.OrderInvoice.Quantity) DESC
```

In query 3, we selected top 15 most buy item, and listed with 5 columns, ItemName from item and CategoryName from category table and sum items with code from above like query 2 and show quantity amount from order table and also shows only months like query from above and the result will be shown below.

Result of query 3

	ItemName	CategoryName	Total Price	Quantity	month
1	Jumper Suits	Halcyon	3600000.00	90	October
2	Coat	Accessoires par Charade	4550000.00	70	October
3	Tshirt	Jumpsync	200000.00	10	October
4	Belts and Ties	Oracle	200000.00	10	October
5	Jumper	Paradox	189000.00	9	October
6	Coat	Accessoires par Charade	455000.00	7	October
7	Play Suits	Temptation	240000.00	6	October
8	Jackets	Virtue	180000.00	2	October
9	Jeans	Illume	30000.00	1	October

Query 4

```

---Query 4---
declare @FromMonth datetime
declare @ToMonth datetime
set @FromMonth = '20220901'
set @ToMonth = '20221231'
select dbschema.Customer.CustomerName, count(dbschema.Customer.CustomerID) as "Total Ordering transaction", datename(month, dbschema.Ord.OrderDate) as Month
from dbschema.Customer inner join dbschema.Ord on dbschema.Customer.CustomerID = dbschema.Ord.CustomerID
where dbschema.Ord.OrderDate between @FromMonth and @ToMonth
group by datename(month, dbschema.Ord.OrderDate), dbschema.Customer.CustomerID, dbschema.Customer.CustomerName
order by dbschema.Customer.CustomerID ASC

```

In this query 4, it shows which customer is on transaction and if item is on the way for delivery, the transaction will show 0 and the result will show below.

Result of query 4

	CustomerName	Total Ordering transaction	Month
1	Selena	1	October
2	Maya	1	October
3	Maya	1	September
4	Billie	2	October
5	Billie	1	September
6	Taylor	1	November
7	Taylor	1	September
8	Polly	1	October
9	Corbyn	1	October
10	Don	1	October
11	Thelma	1	October

Query 5

```

---Query 5 ---
select dbschema.Ord.OrderDate as Date, sum(dbschema.Item.ItemUnitPrice * dbschema.OrderInvoice.Quantity)
as "Total Selling amount", sum(dbschema.Ord.Discount) as "Total Discount amount", sum(dbschema.Ord.DeliveryFee) as "Total Delivery Charges"
from dbschema.Ord inner join dbschema.OrderInvoice on dbschema.Ord.OrderNumber = dbschema.OrderInvoice.OrderNumber inner join dbschema.Item
on dbschema.OrderInvoice.ItemID = dbschema.Item.ItemID
group by dbschema.Ord.OrderDate

```

In this query, it shows all full date including days, months and years as delivery date with total selling amount and total discount amount with coding above and it also shows total delivery charges too. The quantity and item unit price are added to determine the total selling amount. Select the order date. The result will be shown below.

Result in query 5

	Date	Total Selling amount	Total Discount amount	Total Delivery Charges
1	2022-09-20	9350000.00	0	14000.00
2	2022-10-03	30000.00	0	4000.00
3	2022-10-04	180000.00	0	2000.00
4	2022-10-10	189000.00	0	4000.00
5	2022-10-13	3600000.00	0	3000.00
6	2022-10-14	200000.00	0	10000.00
7	2022-10-15	240000.00	0	4000.00
8	2022-10-18	200000.00	0	2000.00
9	2022-10-21	4550000.00	0	2000.00
10	2022-10-22	455000.00	0	3000.00
11	2022-11-02	360000.00	0	3000.00

Query 6

```
--Query 6 ----
select top 5 dbschema.SF.SFID, dbschema.SF.SFName, count(dbschema.SF.SFName) as "Total Ordering count"
from dbschema.SF inner join dbschema.Ord on dbschema.SF.SFID =dbschema.Ord.SFID
where month(dbschema.Ord.OrderDate) = month(getdate())
group by dbschema.SF.SFID, dbschema.SF.SFName
order by count(dbschema.SF.SFID) DESC
```

In this query, it shows top 5 data from Staff table with SFID and SFName and count as "Total Ordering Count". The result will be show below. **Result of query 6**

	SFID	SFName	Total Ordering count
1	SF13	Shaw	4
2	SF16	West	2
3	SF14	Simond	1

Part 2

Log in

```
---create login with password ---  
CREATE Login AdminofDDD  
with PASSWORD = 'sam123';  
  
---create admin user---  
CREATE user adminDDD for Login AdminofDDD;  
  
---create admin schema--  
---create schema ADMINISTRATOR ;---
```

This is log in page for user and both user and employee in the company can log and access the database with given passwords shown in above coding. And we create one user profile per person.

Admin grant access

```
--grant access on adminDB--  
grant select on dbschema.Category to adminDDD;  
grant select on dbschema.Item to adminDDD;  
grant select on dbschema.Ord to adminDDD;  
grant select on dbschema.OrderInvoice to adminDDD;  
grant select on dbschema.SF to adminDDD;  
grant select on dbschema.Customer to adminDDD;  
grant insert on dbschema.Item to adminDDD;
```

Messages
Commands completed successfully.
Completion time: 2022-10-22T20:07:07.7815117+06:30

This is the grant access of the user access. User can view category, item, order, order in voice, staff, customer name and items name. User can access all these tables above according to the login access and view their order list and details.

Stuff login

```
---employee login--  
CREATE Login EmployeeofDDD  
with PASSWORD = 'sam321';  
  
--create staff user ---  
CREATE user staffDDD for Login EmployeeofDDD;
```

This is log in page for employee only in the company can log and access the database with given passwords shown in above coding. And we create one employee profile per person. Employee can access user profile but user cannot access the employee access because it is for company employee only.

Stuff grant access

```
--grant acces on staffDDD--  
grant select on dbschema.Category to staffDDD;  
grant select on dbschema.Ord to staffDDD;  
grant select on dbschema.OrderInvoice to staffDDD;  
grant select on dbschema.SF to staffDDD;  
grant select on dbschema.Customer to staffDDD;  
grant insert on dbschema.Item to staffDDD;  
  
--deny delete alter update--
```

This is the grant access of the employee access. User can view category, order, order in voice, staff, customer name and items name. Employee can access all these tables above according to the login access and view their order list and details. And edit the database.

User cannot delete data from the tables and database like category and items and this is the deny, delete, alter and update coding from above and result will be shown below.

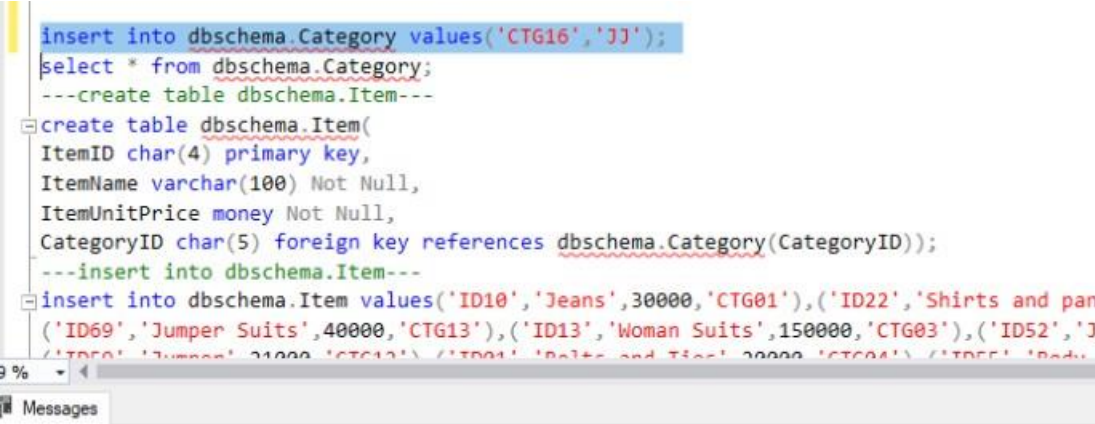
Execute user and testing

```
--deny delete, alter , update---
deny delete,update,alter on dbschema.Category to staffDB;
deny delete, update,alter on dbschema.Item to staffDB;
select * from dbschema.Item;
delete from dbschema.Item where ItemID='ID022';
select * from dbschema.Item;

---execute user---
EXECUTE AS USER = 'StaffDDD'
REVERT
select * from dbschema.Item

---testing---
delete from dbschema.Item where ItemID='ID022';
select * from dbschema.Item;
```

Test plan Category ID test 1



```
insert into dbschema.Category values('CTG16','JJ');
select * from dbschema.Category;
---create table dbschema.Item---
create table dbschema.Item(
  ItemID char(4) primary key,
  ItemName varchar(100) Not Null,
  ItemUnitPrice money Not Null,
  CategoryID char(5) foreign key references dbschema.Category(CategoryID));
---insert into dbschema.Item---
insert into dbschema.Item values('ID10','Jeans',30000,'CTG01'),('ID22','Shirts and pan
('ID69','Jumper Suits',40000,'CTG13'),('ID13','Woman Suits',150000,'CTG03'),('ID52','J
('ID50','Jumper',21000,'CTG13'),('ID01','Belt and Ties',20000,'CTG04'),('ID55','Beds

(1 row affected)

Completion time: 2022-10-22T20:39:21.1024430+06:30
```

we insert values in "DDD.category" from category table and this is the error message due to duplication.

Testing category table with null and not null.

Category ID test 2

```

insert into dbschema.Category values('CTG16','JJ');
insert into dbschema.Category values('CTG016','JJ');
select * from dbschema.Category;
---create table dbschema.Item---
create table dbschema.Item(
  ItemID char(4) primary key,
  ItemName varchar(100) Not Null,
  ItemUnitPrice money Not Null,
  CategoryID char(5) foreign key references dbschema.Category(CategoryID));
---insert into dbschema.Item---
insert into dbschema.Item values('ID10','Jeans',30000,'CTG01'),('ID22','Shirts and pants',70000,'CTG02'),('ID50',
('ID60','Human Suits',40000,'CTG13'),('ID13','Human Suits',150000,'CTG03'),('ID53','Jackets',20000,'CTG11'),('
%
Messages
Msg 2628, Level 16, State 1, Line 29
String or binary data would be truncated in table 'db.dbschema.Category', column 'CategoryID'. Truncated value: 'CTG01'.
The statement has been terminated.

Completion time: 2022-10-22T20:40:20.5696103+06:30

```

We insert values in “DDD.category” from category table and we put new values with wrong data type so this error is shown in above.

Category ID test 3

```

insert into dbschema.Category values('CTG16','JJ');
insert into dbschema.Category values('CTG016','JJ');
insert into dbschema.Category values(not null,'JJ');
select * from dbschema.Category;
---create table dbschema.Item---
create table dbschema.Item(
  ItemID char(4) primary key,
  ItemName varchar(100) Not Null,
  ItemUnitPrice money Not Null,
  CategoryID char(5) foreign key references dbschema.Category(CategoryID)
---insert into dbschema.Item---
insert into dbschema.Item values('ID10','Jeans',30000,'CTG01'),('ID22',
%
Messages
Msg 156, Level 15, State 1, Line 30
Incorrect syntax near the keyword 'not'.

Completion time: 2022-10-22T20:40:49.0321692+06:30

```

We insert values in “DDD.category” from category table and we put new values with not null and this error is shown in above.

No	Purpose	Key/Constraint	Data type	Test Data	Description	Expected Result	Actual Result	Pass/Fail
1	To test the Section ID	Primary Key	Char (5) not null	CTG16	To save the Section ID	For the Database to Accept the Section ID	Accept the section ID state	Pass
2	To test the Section ID	Primary Key	Char (5) not null	CTG016	To save the Section ID	For the Database to Accept the Section ID	Reject the section Id state because of data duplication	fail
3	To test the Section ID	Primary Key	Char (5) not null	Not null	To save the Section ID	For the Database to Accept the Section ID	Reject the section Id state because of wrong format	fail

Item ID Test 1

```

('ID58','Bloomer',45000,'CTG07'),('ID57','Coat',65000,'CTG06'),('ID56','Play Suits'
insert into dbschema.Item values('ID71','leather',8000,'CTG01')
select * from dbschema.Item

insert into dbschema.Item values('ID69','red coat',10000,'CTG01')
insert into dbschema.Item values('ID069','leather',10000,'CTG01')
insert into dbschema.Item values('ID071','leather',10000,'CTG01')

```

9 %

Messages

(1 row affected)

(1 row affected)

Completion time: 2022-10-22T20:33:18.8266775+06:30

We insert right data type to “DDDsche.Item” from item table within “DDDsche” schema and result is from above coding.

Item ID Test 2

```

insert into dbschema.Item values('ID71','red coat',10000,'CTG01')
insert into dbschema.Item values('ID071','leather',10000,'CTG01')

select * from dbschema.Item

```

%

Messages

Msg 2628, Level 16, State 1, Line 44
String or binary data would be truncated in table 'db.dbschema.Item', column 'ItemID'. Truncated value: 'ID071'.
The statement has been terminated.

Completion time: 2022-10-22T20:34:58.7237879+06:30

We insert wrong data to same table from item ID test 1 by putting “0” in the middle for 5 words and the result shown in above coding.

Item ID Test 3

```

insert into dbschema.Item values('ID71','red coat',10000,'CTG01')
insert into dbschema.Item values('ID071','leather',10000,'CTG01')
insert into dbschema.Item values(not null,'leather',10000,'CTG01')

select * from dbschema.Item

```

09 %

Messages

Msg 156, Level 15, State 1, Line 45
Incorrect syntax near the keyword 'not'.

Completion time: 2022-10-22T20:35:07.5810761+06:30

We insert not null type data to same table from item ID test 1 and 2 and result are from above coding.

No	Purpose	Key/Constraint	Data type	Test Data	Description	Expected Result	Actual Result	Pass/Fail
1	To test the Section ID	Primary Key	Char (4) not null	ID71	To save the Section ID	For the Database to Accept the Section ID	Accept the section ID state	Pass
2	To test the Section ID	Primary Key	Char (4) not null	ID071	To save the Section ID	For the Database to Accept the Section	Reject the section Id state because of data	fail

						ID	duplication	
3	To test the Section ID	Primary Key	Char (4) not null	Not null	To save the Section ID	For the Database to Accept the Section ID	Reject the section Id state because of wrong format	fail

Staff ID Test 1

```

insert into DDDsche.SF values ('SF17','Tom')
select * from DDDsche.SF
---create table DDDsche.Cus---
create table DDDsche.Cus(
  CustomerID char(5) primary key,
  CustomerName varchar(100) Not Null)

```

1 %

Messages

(1 row affected)

Completion time: 2022-10-22T11:09:29.7290177+06:30

we insert right data type to "DDDsche.SF" from staff table with "DDDsche" schema and result are shown above.

Staff ID Test 2

```
insert into DDDsche.SF values ('SF17','Tom')
insert into DDDsche.SF values ('SF017','Tom')
select * from DDDsche.SF

---create table DDDsche.Cus---
create table DDDsche.Cus(
  CustomerID char(5) primary key,
```

1 %

Messages

Msg 2628, Level 16, State 1, Line 61
String or binary data would be truncated in table 'SAM2.DDD.SF', column 'SFID'. Truncated value: 'SF01'.
The statement has been terminated.

Completion time: 2022-10-22T11:10:02.1534707+06:30

We insert wrong data type by putting "0" to same table from staff id test and the result are shown above.

Staff ID Test 3

```
insert into DDDsche.SF values ('SF17','Tom')
insert into DDDsche.SF values ('SF017','Tom')
insert into DDDsche.SF values (not null,'Tom')
select * from DDDsche.SF

---create table DDDsche.Cus---
create table DDDsche.Cus(
```

21 %

Messages

Msg 156, Level 15, State 1, Line 62
Incorrect syntax near the keyword 'not'.

Completion time: 2022-10-22T11:11:08.1275482+06:30

We insert not null data to same table from staff id test 1 and 2 and result are shown above.

No	Purpose	Key/Constraint	Data type	Test Data	Description	Expected Result	Actual Result	Pass/Fail
1	To test the Section ID	Primary Key	Char (4) not null	SF17	To save the Section ID	For the Database to Accept the Section ID	Accept the section ID state	Pass
2	To test the Section ID	Primary Key	Char (4) not null	SF017	To save the Section ID	For the Database to Accept the Section ID	Reject the section Id state because of data duplication	fail
3	To test the Section ID	Primary Key	Char (4) not null	Not null	To save the Section ID	For the Database to Accept the Section ID	Reject the section Id state because of wrong format	fail

Customer ID Test 1

```

CustomerName varchar(100) Not Null)

insert into dbschema.Customer values ('CUS17','heather')

select * from dbschema.SF

---insert into dbschema.Customer---
insert into dbschema.Customer values('CUS01','Selena');
insert into dbschema.Customer values('CUS04','Mavis') ('CUS06','Bella') ('C

109 %
Messages

(1 row affected)

Completion time: 2022-10-22T20:37:32.0158590+06:30

```

We insert the right values data type in "DDDsche.Cus" from customer table and the result are shown above.

Customer ID Test 2

```

CustomerID char(5) primary key,
CustomerName varchar(100) Not Null)

insert into dbschema.Customer values ('CUS17','heather')
insert into dbschema.Customer values ('CUS017','heather')

select * from dbschema.SF

---insert into dbschema.Customer---
insert into dbschema.Customer values('CUS01','Selena');

109 %
Messages

Msg 2628, Level 16, State 1, Line 73
String or binary data would be truncated in table 'db.dbschema.CUS0mer', column 'CUS0merID'. Truncated value: 'CUS01'.
The statement has been terminated.

Completion time: 2022-10-22T20:38:33.3075807+06:30

```

We insert wrong data type to same table as customer id test 1 and the result are shown above.

Customer ID Test 3

The screenshot shows a MySQL command window with the following SQL commands:

```

CREATE TABLE dbschema.Customer (
  CustomerID char(5) primary key,
  CustomerName varchar(100) Not Null)

insert into dbschema.Customer values ('CUS17','heather')
insert into dbschema.Customer values ('CUS017','heather')
insert into dbschema.Customer values (not null,'heather')
select * from dbschema.SF

---insert into dbschema.Customer---
insert into dbschema.Customer values ('CUS01','Seless');
  
```

The error message displayed is:

```

Msg 156, Level 15, State 1, Line 74
Incorrect syntax near the keyword 'not'.

Completion time: 2022-10-22T20:38:37.1354472+06:30
  
```

We insert not null to same table as customer id test 1 and 2 and results are shown above.

No	Purpose	Key/Constraint	Data type	Test Data	Description	Expected Result	Actual Result	Pass/Fail
1	To test the Section ID	Primary Key	Char (4) not null	CUS17	To save the Section ID	For the Database to Accept the Section ID	Accept the section ID state	Pass

2	To test the Section ID	Primary Key	Char (4) not null	CUS017	To save the Section ID	For the Database to Accept the Section ID	Reject the section Id state because of data duplication	fail
3	To test the Section ID	Primary Key	Char (4) not null	Not null	To save the Section ID	For the Database to Accept the Section ID	Reject the section Id state because of wrong format	fail
	ID					Section ID		

Part 3

Backup and restore

```

assignment2DDD(s...UOSLS\samwi (58))  DDDAssignment2(s...UOSLS\samwi (68))* -p X
select * from DDDsche.Item;

--dinstinction--
--backup plan database--
backup database DDDsche
To disk='C:\HND43\database\assignment2.bak'
--Restore database'--
restore database assignment2
from disk = 'C:\HND43\database\assignment2.bak' with replace;

---Create Trigger---
Create table DDDsche.ordtrigger(
    id int IDENTITY(1,1) Primary key,
    Description varchar (100));

create trigger dbordtriger
on DDDsche.Ord
after insert
  
```

91 %

Results Messages

	id	Description
1	1	One row is insertedOct 21 2022 6:23PM
2	2	One row is insertedOct 21 2022 6:24PM

We create backup and restore database just in case if file got deleted or something happens and coding will not be same due to different location in employee's pc. **Trigger**

Order Trigger

```

---Create Trigger---
Create table DDDsche.ordtrigger(
    id int IDENTITY(1,1) Primary key,
    Description varchar (100));
  
```

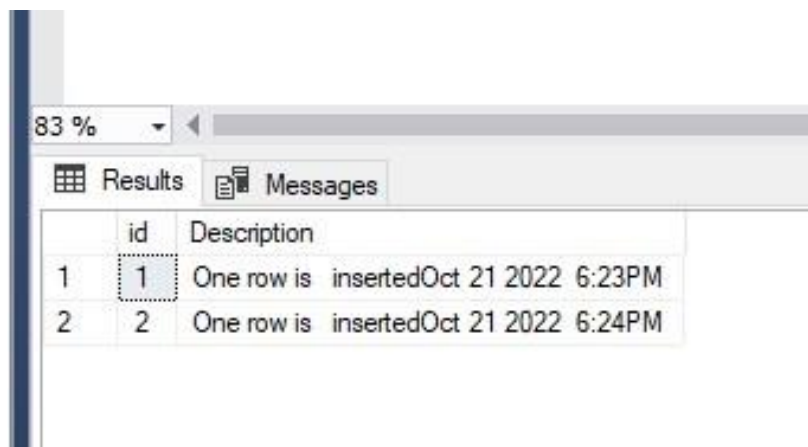
We create tale for trigger and name it "DDDsche.ordTrigger" under the "DDDsche" shema and insert 2 columns. The first one is "id" as primary key and second one is "description" with "varchar" data type with "100" words allowance.

```
create trigger dbordtriger
on DDDsche.Ord
after insert
as
begin
insert into DDDsche.ordtrigger values (CONCAT('One row is
inserted',GETDATE()))
end

insert into DDDsche.Ord values ('0031','2022/9/10','2022/9/12',2000,0,'9:15','$1.5','SF05','CS011')
select * from DDDsche.Ord
select * from DDDsche.ordtrigger
insert into DDDsche.Ord values ('0032','2022/9/10','2022/9/12',2000,0,'9:15','$1.5','SF05','CS011')
```

We create trigger and name it "dbordTrigger" on "DDD.ord" from order table and use "after insert as begin" code and insert "DDDsche.ordtrigger". Insert values into "DDDsche.ord" from order table and result will be shown below.

Result of order trigger



The screenshot shows the SQL Server Enterprise Manager interface. The 'Results' window is active, displaying two rows of data. The first row has an ID of 1 and a description 'One row is insertedOct 21 2022 6:23PM'. The second row has an ID of 2 and a description 'One row is insertedOct 21 2022 6:24PM'. The 'Messages' window is also visible but empty.

	id	Description
1	1	One row is insertedOct 21 2022 6:23PM
2	2	One row is insertedOct 21 2022 6:24PM

Item trigger

```
---Create Trigger---
Create table DDDsche.Itemtrigger(
  id int IDENTITY(1,1) Primary key,
  Description varchar (100));

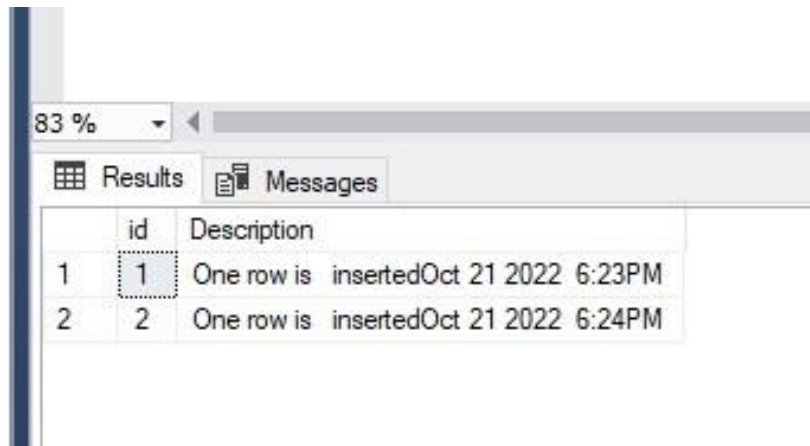
create trigger ITtrigger
on DDDsche.Item
after insert
as
begin
  insert into DDDsche.ITtrigger values (CONCAT('One row is
inserted',GETDATE()))
end

insert into DDDsche.Item values ('ID02','Ring',1000,'CS001')
insert into DDDsche.Item values ('ID69','manSuit',20000,'CS009')
select * from DDDsche.Item
select * from DDDsche.Itemtrigger
```

This is same as order table from above with the difference of values. We create table for item trigger within “DDDsche” schema and name it “DDDsche.Itemtrigger” with 2 columns same as order table with same data type and keys.

We create trigger and name it “ITtrigger” on “DDDsche.Item” and use “after insert begin” and insert values. We insert values to “DDDsche.Item” from item table and result will be shown below.

Result of item trigger



	id	Description
1	1	One row is insertedOct 21 2022 6:23PM
2	2	One row is insertedOct 21 2022 6:24PM

Assess the effectiveness of the testing, AC an explanation of the choice of test data used.

A key component of implementing IT systems that power business applications and deliver analytical data to support operational decisionmaking and strategic planning by corporate executives, business managers, and other end users is effective data management.

Trigger

A database trigger is a piece of procedural code that executes automatically in response to specific events on a specific table or view. The trigger primarily serves to preserve the accuracy of the database's information.

A trigger is a specific kind of stored procedure that launches automatically whenever a database server event takes place. When a user attempts to edit data using a data manipulation language (DML) event, DML triggers are activated. DML operations are statements that INSERT, UPDATE, or DELETE data from a table or view.

Conclusion

In this report, blue smart clothing company has now full database design and program for the company as in ERD, flowchart, normalization with excel and SQL server database. The company can edit and change data easily and if any further error occurs, they can check from test plan or call us for issues.

Reference

<https://www.geeksforgeeks.org/sql-trigger-student-database/>

<https://effectivedatabase.com/>
