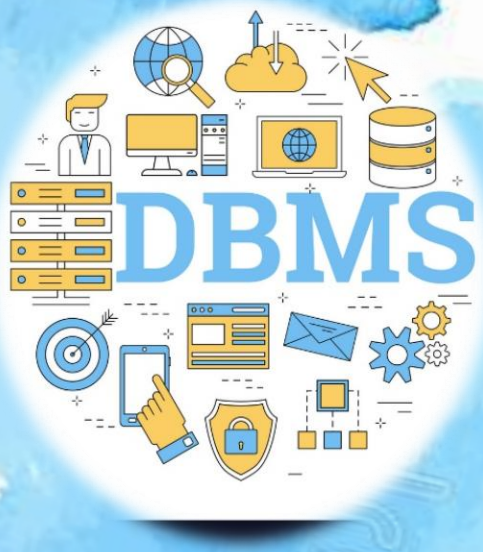
UNIVERSITY OF ECONOMICS AND LAW

Faculty of Information Systems

Project

SALES DATA MANAGEMENT OF ANO COMPANY



Member of ANO Group:

Mai Nguyễn An Bình

Hoàng Nguyễn Quỳnh Hương

Trương Thị Tuyết Nga

TP. HCM, June 2020

MEMBER OF ANO GROUP

No.	Full name	Student code	Role	Phone	Email
1	Mai Nguyễn An Bình	K184060777	Leader	0853103355	binhmna18406c@st.uel.edu.vn
2	Hoàng Nguyễn Quỳnh Hương	K184060789	Member	0918519989	huonghnq18406c@st.uel.edu.vn
3	Trương Thị Tuyết Nga	K184060795	Member	0933409829	ngattt18406c@st.uel.edu.vn

TABLE OF CONTENTS

CHAPTER 1: OVERVIEW OF THE TOPIC

1.Reasons to choose

Vietnam's economy is on the way to integrate with the regional and world economy, the business environment of enterprises has been expanded, but the competition is also becoming more fierce. An enterprise that wants to do business effectively, must exchange with the market. Therefore the question of how to manage the sales process takes place effectively and achieves the highest benefit.

Businesses want to operate effectively must always require great efforts from administrators in the management and exploitation of information through forms, statistical reports, databases,… From activities such as self-assessment, situation forecasting to strategic planning, making the final decision to develop a development strategy, the Database plays an important role to support the management.

From the explosion of Information Technology (IT) today, especially the current 4.0 technology revolution, there are many tools to support management database such as Microsoft SQL Server, Oracle, Microsoft Access, PostgreSQL,... It has provided a working environment between users and the database through data access and query operations. There is decentralization, security, and limited exploitation on the database to solve the problems of sharing and disputes on data when many people access it.

From the awareness of the importance of database in my group's business, my group chose the topic: "Sales Data Management of ANO Company" to bring theoretical knowledge to the basic subject database and respond to business practices.

2. Objective - Research method

2.1 Objective

Use Microsoft SQL Server effectively to build, design, and exploit data to meet analysis needs. Also combined with some Business Intelligence tools to aggregate and analyze charts and tables. The control panel helps to visualize data, provides an overview of the details of production operations easily, so that administrators can be proactive, plan responses to system performance alerts.

2.2 Research method

Method of document research: Refer to your graduation thesis. Studying documents "Nhập môn Cơ sở dữ liệu - M.S. Nguyen Duy Nhat", "Khai phá dữ liệu trong kinh doanh - M.S. Nguyen Duy Nhat".

Methods of consultation: Consult data from experts and lecturers to support timely orientation and guidance during the research process.

The method sets out the requirements, hypotheses, and difficulties that businesses may encounter in the future.

Methods to build the Data Warehouse, build SQL Queries, and build charts to explore and visualize data.

Self-study and self-research method for effective use of data visualization tools and the ability to solve business requirements.

3. Technology applied

To do this, the team applied related technologies to design and analysis.

- Microsoft SQL Server: perform queries and requirements that businesses set for the design, construction of Database as well as the resolution of those requirements.

- Business Intelligence tools to visualize data (PivotTable): visualize data tables, which are created after the implementation of enterprise information requests.

- Design tools (Paint): create a demo interface for applications for data management and the requirements set for the process of designing and building interfaces in the future.

4. Related fields - Information needs

4.1 Related fields

In this topic, the team addresses 5 related fields.

- Managing sales data is a professional activity focusing on the practical application of sales techniques and managing the sales activities of a company. This is an important business function such as net sales through the sale of products and services, and as a result, profit drives most of the commercial business.

- Managing financial-accounting data is one of the important jobs of a business manager because good financial management not only helps businesses maximize profits but also moves them forward. From the perspective of today's economists, there are usually two basic goals: Maximize profit after tax, Maximize profit on equity.

- Manage warehouse data to closely monitor goods in the warehouse, rearrange the warehouse, avoid loss of goods, ensure storage standards on several regulations on warehouse safety. Inventory management facilitates easy and error-free periodic inventory reporting, giving administrators an overview of the inventory situation.

- Manage the system to effectively manage employees, closely monitor employee activities, manage time and productivity, work of each employee. Besides, the system administrator helps employees and administrators carry out notification and reminder requests in the company.

- Customer relationship management (CRM) helps to know a lot about their customers and can provide them with positive experiences and bring good results. Sales, marketing and customer service teams can share valuable information about customers to transform them deep down the sales funnel. With this ease, teams can work seamlessly together to improve sales and achieve work efficiency.

4.2 Information needs

In this topic, ANO Group has the following industry-related needs.

- Sales data needs information on invoices, quantities sold, time to create invoices, employees creating invoices, invoice values, ...

- Financial data needs information about the amount of capital generated in the production and business processes of the company, the costs spent to buy products, and operate the system, ...

- Warehouse data needs information about the supplier, price, quantity of inventory, time of inventory, and price fluctuations of that product, ...

- System data needs information on name, title, operation history, and operational requirements on management software, ...

- Customer relationship management data about the company's VIP customers information, offer incentives to that customer, conduct research, and conduct product and service promotion for them…

5. The structure of the report

The content of the report is composed of 5 chapters:

Chapter 1: Overview Of The Topic

Chapter 2: Building And Designing Databases

Chapter 3: Exploiting Information From Databases

Chapter 4: Building Management Application

Chapter 5: Conclude

CHAPTER 2: BUILDING AND DESIGNING DATABASES

1. Survey the current status of business and determine requirements

1.1 Introduction about the organization

ANO is a small company in the center of Ho Chi Minh City. There are many types of milk such as milk powder, yogurt, or canned milk from big distributors in Viet Nam. Despite lately existing, ANO Company quickly becomes one of the most famous companies in that area because of the quality of products and suitable costs. Due to efficient management and higher profit, ANO company needs to change from manual management to a database management system.

State of information technology:

- Hardware: Each department has its computers to serve the functions.
- Software: ANO uses Excel to import manually data every day. Whenever the management department needs reports to analyze their performance, the rest must prepare for many days or even a week.

1.2 Organizational structure

- Management department: include 3 managers who responsible for coordinating the operation of ANO Company.
- Financial-accounting department: play a role as a cashier while record the number of sales.
- Warehouse Department: checking to ensure the quality of goods, managing the warehouse process.
- System department: preserving the store's security.

1.3 Professional activities

The main professional activities at ANO Company such as sales process, warehouse process, financial-accounting management, System management, etc…Professional activities

1.3.1 Sales process

- Customers select what products they will buy.
- The staff creates a bill and receive payment of customers.

1.3.2 Warehouse process

- The leader of warehouse department announce the request to import the missing items from vendors.
- The warehouse keeper compares the quantity of imported items with their purchase orders, then receive invoice from the vendors.
- The accountant makes a warehouse receipt after checking it again.
- The warehouse department complete warehouse process and update on their system.
- If the price of imported goods is different from the stocks, the price will be recalculated using the weighted average method. For instance, a (dong) is an old price of Vinamilk yogurt with 100 products in the warehouse, and b (dong) is a new price in the recent import with 200 products. So the price now is (a*100+b*200)/300 (dong).

1.3.3 Customer management

Financial Accounting department is responsible for recording all transaction information. So that, the managers will send birthday gifts to their customers or have a sale program for customers buying a lot of products every year.

1.4 Accounting management:

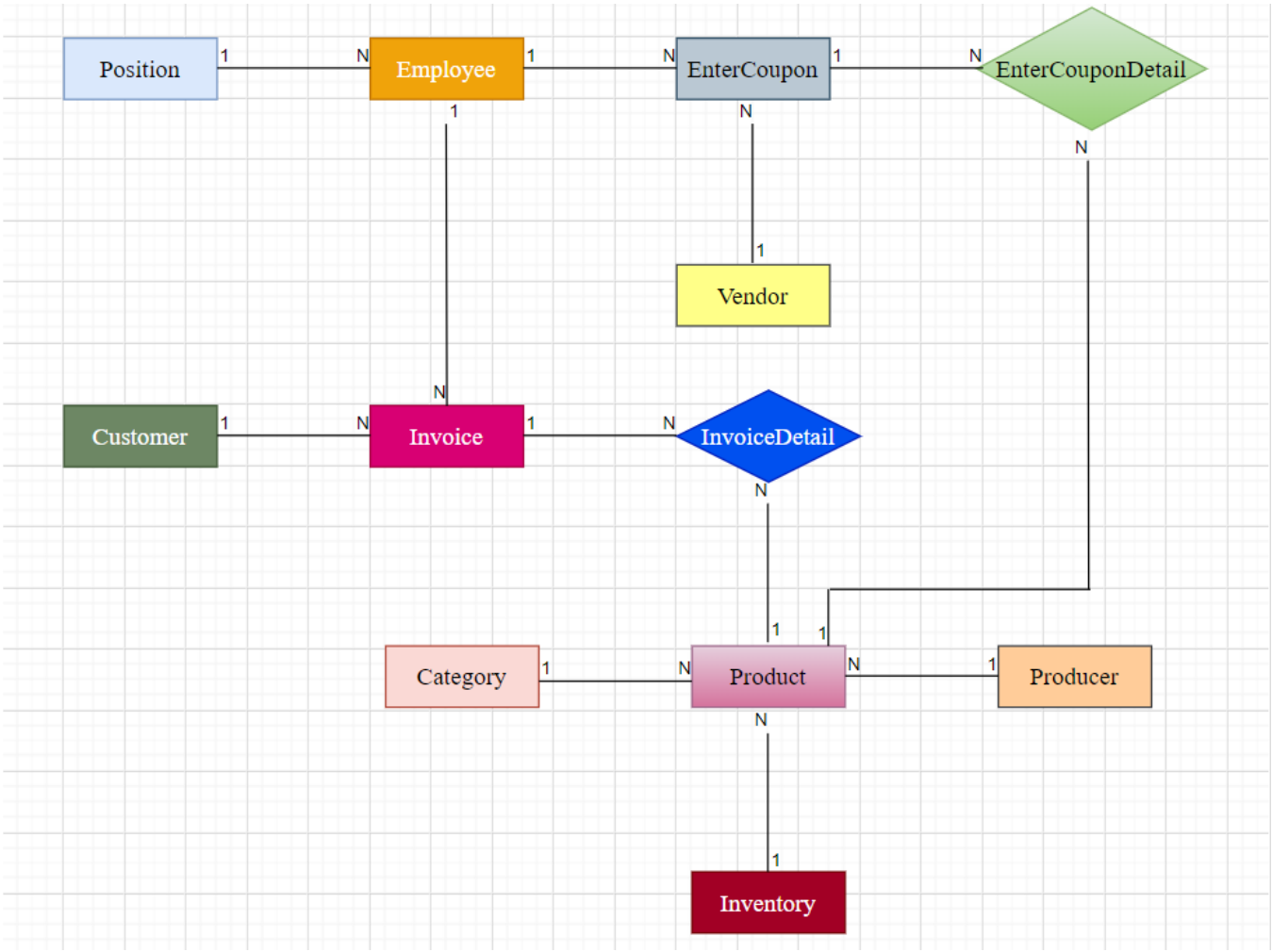
- Cashier is the person who directly deals with cash with customers. These transactions must be fully recorded on the system to help the managers control easily. At the end of the day, the cashier must recalculate the revenue then compare with the cash at store.
- If the payment request is received, the cashier must make a clear payment slip and archive it and similar with collection.

1.5Data request

Operations	**Request**
Sales Management	Create a list of invoice in month x year y
	Calculate revenue and profits of months of year x. In which month do you have the highest revenue?
	Create a list of products sold, total sold, revenue and profit in month x year y
	Top 5 the most sold products in the month and in the year x
	Create list of how much revenue and profit each customer bringing in each month in year x
	Revenue of each producer by month in year x
	Revenue of each category by month in year x
	For the same category (such as powdered milk), which producer & products are the best selling in year x?

	With the x-type category, which ones do customers buy with x
	Check the reliability of this statement: If customers buy the type category x and the y type category at the same time, they will usually buy the z-category
	Calculate the revenue of each item in month x yeary
	Find out which invoice have most items in month x, year y
	Create a list of items which are not purchased in year x
	Make a list of invoices issued from day x to day y
	Insert, delete, edit data
Warehouse Management	Create a list of imported items in month x year y
	Create a list of the transaction with y vendor in month x year y
	Test the reliability of the statement: "Customers tend to buy all items of the same supplier"
	Create a list of items from x (dong) to y (dong), followed by z producer
	Insert, delete, edit data
Financial-Accounting Management	Statistics include the warehousing prices for the months of that year
	Average of invoices for each month in year x
	Insert, delete, edit data
CRM (Customer Relationship Management)	Updated (5% off) on total bill for whose birthdays are in the month
	Among the 5 customers with the highest sales, find customers with the highest number of purchases each month of the year
	Insert, delete, edit data
System Management	Insert, delete, edit data

2. Entity and Relationship Diagram (ERD)



Database schema

Describe: *The underlined attributes are primary keys.*

1. Customer(ID_Customer, NameCustomer, AddressCustomer, PhoneCustomer, BirthdayCustomer, GenderCustomer)
2. Employee(ID_Employee, ID_Position, NameEmployee, PhoneEmployee)
3. Position(ID_Position, Position, Permision)
4. Product(ID_Product, ID_ProductClass, ID_Producer, NameProduct, Price_Purchase, Price_Sale, TotalQuanity, UpdateAt)
5. Category(ID_ProductClass, Category)
6. Producer(ID_Producer, NameProducer)
7. Vendor(ID_Vendor, NameVendor, AddressVendor, PhoneVendor)
8. Invoice(ID_Invoice, ID_Customer, CreateAt, ID_Employee, TotalValue)
9. InvoiceDetail(ID_Invoice, ID_Product, Quanity, TotalValue)
10. EnterCoupon(ID_Enter, ID_Employee, ID_Vendor, CreateAt, TotalValue)
11. EnterCouponDetail(ID_Enter, ID_Product, Quanity, TotalValue)

12. Inventory(Period, ID_Product, QuanityFirst, TotalQuanityPurchase, TotalQuanitySale, QuanityLast)

1. Customer: An entity consists of attributes

ID_Customer: Customer's code

NameCustomer: Customer's name

AddressCustomer: Customer's address

PhoneCustomer: Customer's phone number

BirthdayCustomer: Customer's birthday

GenderCustomer: Customer's gender

2. Employee: An entity consists of attributes

ID_Employee: Employee's code

ID_Position: Position's code

NameEmployee: Employee's name

PhoneEmployee: Employee's phone number

3. Position: An entity consists of attributes

ID_Position: Position's code

Position: Employee's position

Permision: Employee's permissions

4. Product: An entity consists of attributes

ID_Product: Product's code

ID_ProductClass: Category's code

ID_Producer: Producer's code

NameProduct: Product's name

Price_Purchase: Purchase price

Price_Sale: Sale Price

TotalQuanity: Remain quantity

UpdateAt: Date updated

5. Category: An entity consists of attributes

ID_ProductClass: Category’s code

Category: Category’s name

6. Producer: An entity consists of attributes

ID_Producer: Producer’s code

NameProducer: Producer’s name

7. Vendor: An entity consists of attributes

ID_Vendor: Vendor’s code

NameVendor: Vendor’s name

AddressVendor: Vendor’s address

PhoneVendor: Vendor’s phone number

8. Invoice: An entity consists of attributes

ID_Invoice: Invoice’s code

ID_Customer: Customer’s code

CreateAt: Date created

ID_Employee: Employee’s code

TotalValue: Invoice’s value

9. InvoiceDetail: An entity consists of attributes

ID_Invoice: Invoice’s code

ID_Product: Product’s code

Quanity: Product’s quantity

TotalValue: Product’s total value

10. EnterCoupon: An entity consists of attributes

ID_Enter: Enter coupon’s code

ID_Employee: Employee’s code

ID_Vendor: Vendor’s code

CreateAt: Date created

TotalValue: Enter coupon’s value

11. EnterCouponDetail: An entity consists of attributes

ID_Enter: Enter coupon’s code

ID_Product: Product’s code

Quanity: Product’s quantity

TotalValue: Enter coupon;s total value

12. Inventory: An entity consists of attributes

Period: Period update

ID_Product: Product’s code

QuanityFirst: Beginning quantity

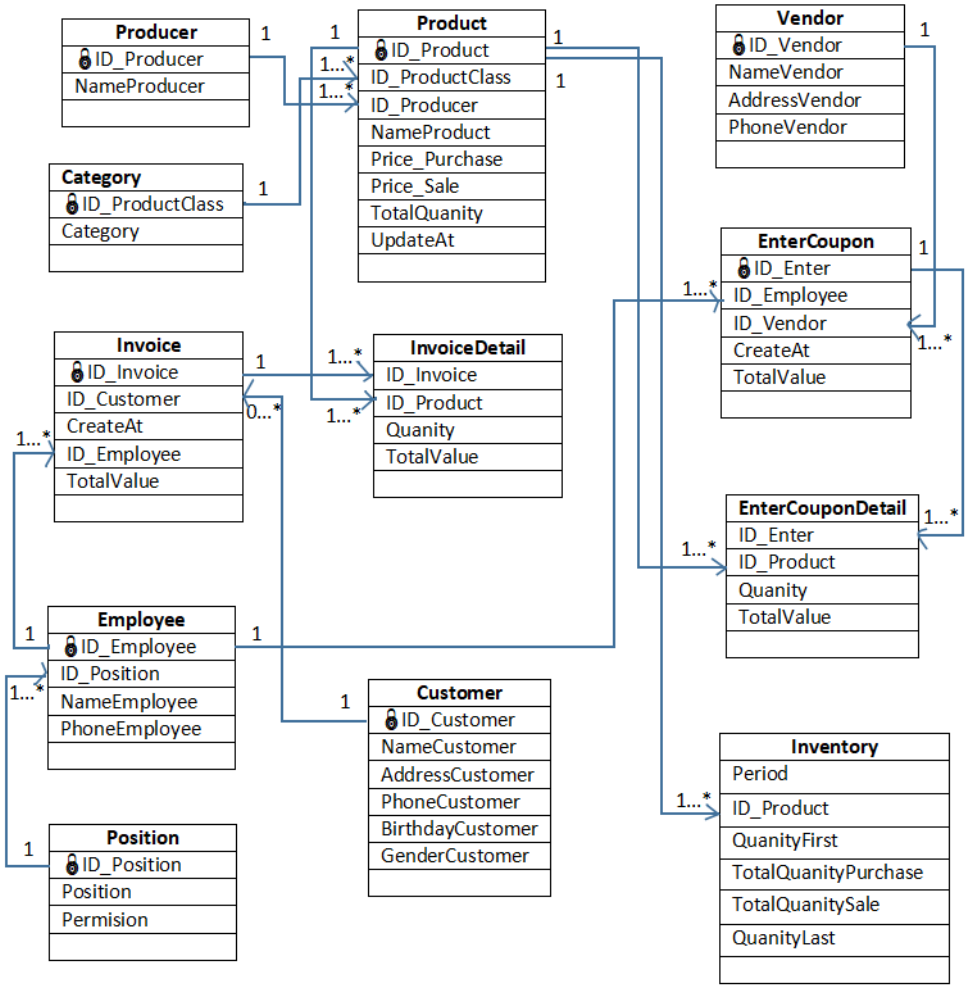
TotalQuanityPurchase: Total product purchased in period

TotalQuanitySale: Total product sold in period

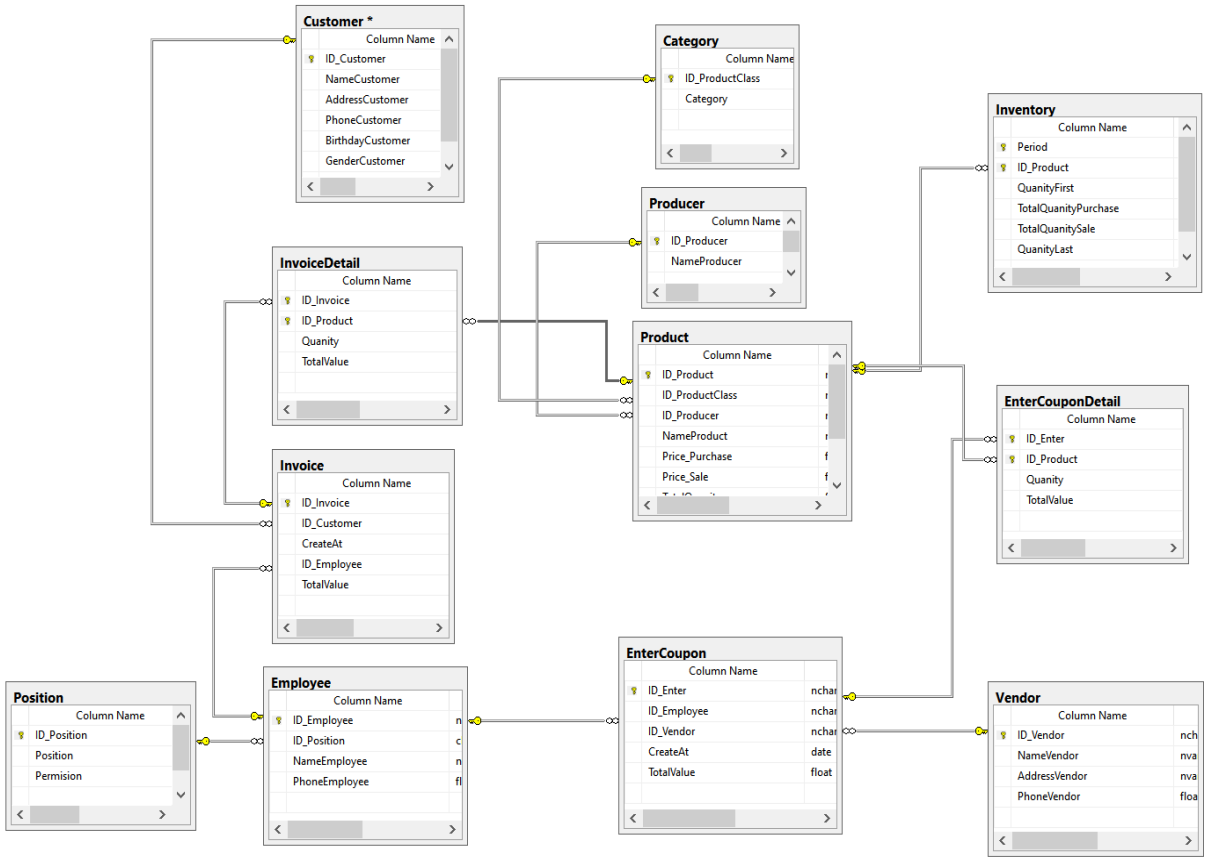
QuanityLast: Ending quantity

3. Logical design for the diagram

3.1 Logical Model



3.2 Logical Diagram



3.3 Describe relationships between entities

1. A producer (Producer) can have many products (Product), a product is just produced by a certain producer.
2. A category (Category) can have many products (Product), a product belongs to only one category.
3. A product (Product) can appear in many invoice details (InvoiceDetail), an invoice detail has a certain product. (1 - ∞)
4. An invoice (Invoice) consists of many invoice details (InvoiceDetail), an invoice detail is only contained in certain invoice. (1 - ∞)
5. A customer (Customer) can have many invoices (Invoice), an invoice belongs to only one customer.(1 - ∞)
6. An employee (Employee) can create many invoices (Invoice), an invoice is created by only one certain employee. (1 - ∞)
7. A position (Position) can belong to many employees (Employee), an employee only have a position in company. (1 - ∞).
8. A vendor (Vendor) can have many enter coupons (EnterCoupon), an enter coupon just belongs to only one certain vendor. (1 - ∞)

9. An enter coupon (EnterCoupon) consists of many enter coupon details (EnterCouponDetail), an enter coupon detail is only contained in certain enter coupon. (1 - ∞)
10. A product (Product) can appear in many enter coupon details (EnterCouponDetail), an enter coupon detail has a certain product. (1 - ∞)
11. An employee (Employee) can create many enter coupon (EnterCoupon), an enter coupon is created by only one certain employee. (1 - ∞)
    In certain period, many products (Product) are inventoried (Inventory), but inventory is determined over multiple periods. (1 - ∞)

4. Physical design for the diagram

1. Position

Field name	**Data type**	**Description**
ID_Position	char(4)	Position's code
Position	nchar(50)	Employee's position
Permision	nchar(50)	Employee's permissions

2. Employee

Field name	**Data type**	**Description**
ID_Employee	nchar(10)	Employee's code
ID_Position	char(4)	Position's code
NameEmployee	nvarchar(100)	Employee's name
PhoneEmployee	float	Employee's phone number

3. Customer

Field name	**Data type**	**Description**
ID_Customer	nchar(10)	Customer's code
NameCustomer	nvarchar(100)	Customer's name
AddressCustomer	nvarchar(100)	Customer's address
PhoneCustomer	float	Customer's phone number
BirthdayCustomer	date	Customer's birthday
GenderCustomer	nvarchar(10)	Customer's gender

4. Product

Field name	Data type	Description
ID_Product	nchar(10)	Product's code
ID_ProductClass	nchar(10)	Category's code
ID_Producer	nchar(10)	Producer's code
NameProduct	nvarchar(100)	Product's name
Price_Purchase	float	Purchase price
Price_Sale	float	Sale Price
TotalQuanity	float	Remain quantity
UpdateAt	datetime	Date updated

5. Category

Field name	Data type	Description
ID_ProductClass	nchar(10)	Category's code
Category	nvarchar(100)	Category's name

6. Producer

Field name	Data type	Description
ID_Producer	nchar(10)	Producer's code
NameProducer	nvarchar(100)	Producer's name

7. Vendor

Field name	Data type	Description
ID_Vendor	nchar(10)	Vendor's code
NameVendor	nvarchar(100)	Vendor's name
AddressVendor	nvarchar(100)	Vendor's address
PhoneVendor:	float	Vendor's phone number

8. Invoice

Field name	Data type	Description
ID_Invoice	nchar(10)	Invoice’s code
ID_Customer	nchar(10)	Customer’s code
CreateAt	date	Date created
ID_Employee	nchar(10)	Employee’s code
TotalValue	float	Invoice’s value

9. InvoiceDetail

Field name	Data type	Description
ID_Invoice	nchar(10)	Invoice’s code
ID_Product	nchar(10)	Product’s code
Quanity	float	Product’s quantity
TotalValue	float	Product’s total value

10. EnterCoupon

Field name	Data type	Description
ID_Enter	nchar(10)	Enter coupon’s code
ID_Employee	nchar(10)	Employee’s code
ID_Vendor	nchar(10)	Vendor’s code
CreateAt	date	Date created
TotalValue	float	Enter coupon’s value

11. EnterCouponDetail

Field name	Data type	Description
ID_Enter	nchar(10)	Enter coupon’s code

ID_Product	nchar(10)	Product's code
Quanity	float	Product's quantity
TotalValue	float	Enter coupon;s total value

12. Inventory

Field name	**Data type**	**Description**
Period	datetime	Period update
ID_Product	nchar(10)	Product's code
QuanityFirst	float	Beginning quantity
TotalQuanityPurchase	float	Total product purchased in period
TotalQuanitySale	float	Total product sold in period
QuanityLast	float	Ending quantity

5. Integrity constraint

Integrity constraints, specified by formal symbols and the table of effects of that constraint.

5.1 Constraint one relationship

R1: In relationship Producer, every producer has different code

$\forall t_1, t_2 \in$ Producer, $t_1 \neq t_2$, t_1.ID_Producer $\neq t_2$.ID_Producer

R1	Insert	Delete	Edit
Producer	+	-	+(ID_Producer)

R2: In relationship Category, every category has different code

$\forall t_1, t_2 \in$ Category, $t_1 \neq t_2$, t_1.ID_ProductClass $\neq t_2$.ID_ProductClass

R2	Insert	Delete	Edit
Category	+	-	+(ID_ProductClass)

R3: In relationship Vendor, every vendor has different code

$\forall t_1, t_2 \in$ Vendor, $t_1 \neq t_2$, t_1.ID_Vendor $\neq t_2$.ID_Vendor

R3	Insert	Delete	Edit
Vendor	+	-	+(ID_Vendor)

R4: In relationship Product, every product has different code

$\forall t_1, t_2 \in$ Product, $t_1 \neq t_2$, t_1.ID_Product $\neq t_2$.ID_Product

R4	Insert	Delete	Edit
Product	+	-	+(ID_Product)

R5: In relationship Customer, every customer has different code

$\forall t_1, t_2 \in$ Customer, $t_1 \neq t_2$, t_1.ID_Customer $\neq t_2$.ID_Customer

R5	Insert	Delete	Edit
Customer	+	-	+(ID_Customer)

R6: In relationship Invoice, every invoice has different code

$\forall t_1, t_2 \in$ Invoice, $t_1 \neq t_2$, t_1.ID_Invoice $\neq t_2$.ID_Invoice

R6	Insert	Delete	Edit
Invoice	+	-	+(ID_Invoice)

R7: In relationship EnterCoupon, every enter coupon has different code

$\forall t_1, t_2 \in$ Entercoupon, $t_1 \neq t_2$, t_1.ID_Enter $\neq t_2$.ID_Enter

R7	Insert	Delete	Edit
EnterCoupon	+	-	+(ID_Enter)

R8: In relationship Employee, every employee has different code

$\forall t_1, t_2 \in$ Employee, $t_1 \neq t_2$, t_1.ID_Employee $\neq t_2$.ID_Employee

R8	Insert	Delete	Edit
Employee	+	-	+(ID_Employee)

R9: In relationship Position, every position has different code

$\forall t_1, t_2 \in$ Position, $t_1 \neq t_2$, t_1.ID_Position $\neq t_2$.ID_Position

R9	Insert	Delete	Edit
Position	+	-	+(ID_Position)

Property constraints

R10: In relationship Product, one product always has value of attribute Price_Purchase smaller than the one of attribute Price_Sale.

$\forall t \in$ Product $\rightarrow$ t.Price_Purchase < t.Price_Sale

R10	Insert	Delete	Edit
Product	+	-	+(Price_Purchase, Price_Sale)

R11: In relationship InvoiceDetail, quantity of one product always has value greater than 0.

$\forall t \in$ InvoiceDetail $\rightarrow$ t.Quanity > 0

R11	Insert	Delete	Edit
InvoiceDetail	+	-	+(Quanity)

R12: In relationship EnterCouponDetail, quantity of one product always has value greater than 0.

$\forall t \in$ EnterCouponDetail $\rightarrow$ t.Quanity > 0

R12	Insert	Delete	Edit
EnterCouponDetail	+	-	+(Quanity)

R13: In relationship Inventory, ending quantity must equal beginning quantity

sum Total product purchased in period minus Total product sold in period .

∀ t ∈ EnterCouponDetail → t.QuaniyLast = t.QuanityFirst + t.TotalQuanityPurchase - t.TotalQuanitySale

R13	Insert	Delete	Edit
Inventory	+	-	+(QuanityFirst, TotalQuanityPurchase, TotalQuanitySale, QuaniyLast)

5.2 Constraint many relationships

R1: In relationship Product and Producer, attribute ID_Producer of each line in the Product relation must correspond to the Producer relation respectively.

∀ t_1 ∈ Product, ∃ t_2 ∈ Producer: t_1.ID_Producer = t_2.ID_Producer

R1	Insert	Delete	Edit
Product	+	-	+(ID_Producer)
Producer	-	+	-

R2: In relationship Product and Category, attribute ID_ProductClass of each line in the Product relation must correspond to the Vendor relation respectively.

∀ t_1 ∈ Product, ∃ t_2 ∈ Category: t_1.ID_ProductClass= t_2.ID_ProductClass

R2	Insert	Delete	Edit
Product	+	-	+(ID_ProductClass)
Category	-	+	-

R3: In relationship Employee and Position, attribute ID_Position of each line in the Employee relation must correspond to the Position relation respectively.

∀ t_1 ∈ Employee, ∃ t_2 ∈ Position: t_1.ID_Positionr = t_2.ID_Position

R3	Insert	Delete	Edit
Employee	+	-	+(ID_Position)
Position	-	+	-

R4: In relationship Employee and Invoice, attribute ID_Employee of each line in the Invoice relation must correspond to the Employee relation respectively.

$\forall t_1 \in$ Invoice, $\exists t_2 \in$ Employee: t_1.ID_Employee = t_2.ID_Employee

R4	Insert	Delete	Edit
Employee	-	+	-
Invoice	+	-	+(ID_Employee)

R5: In relationship Employee and EnterCoupon, attribute ID_Employee of each line in the EnterCoupon relation must correspond to the Employee relation respectively.

$\forall t_1 \in$ EnterCoupon, $\exists t_2 \in$ Employee: t_1.ID_Employee = t_2.ID_Employee

R5	Insert	Delete	Edit
Employee	-	+	-
EnterCoupon	+	-	+(ID_Employee)

R6: In relationship Customer and Invoice, attribute ID_Customer of each line in the Invoice relation must correspond to the Customer relation respectively.

$\forall t_1 \in$ Invoice, $\exists t_2 \in$ Customer: t_1.ID_Customer = t_2.ID_Customer

R6	Insert	Delete	Edit
Customer	+	-	+(ID_Customer)
Invoice	-	+	-

R7: In relationship Vendor and EnterCoupon, attribute ID_Vendor of each line in the EnterCoupon relation must correspond to the Vendor relation respectively.

$\forall t_1 \in$ EnterCoupon, $\exists t_2 \in$ Vendor: $t_1.$ID_Vendor = t_2.ID_Vendor

R7	Insert	Delete	Edit
Vendor	-	+	-
EnterCoupon	+	-	+(ID_Vendor)

R8: In relationship Inventory and Product, attribute ID_Product of each line in the Inventory relation must correspond to the Product relation respectively.

$\forall t_1 \in$ Inventory, $\exists t_2 \in$ Product: $t_1.$ID_Product = t_2.ID_Product

R8	Insert	Delete	Edit
Inventory	+	-	+(ID_Product)
Product	+	-	+(ID_Product)

R9: In relationship InvoiceDetail and Invoice, attribute ID_Invoice of each line in the InvoiceDetail relation must correspond to the Invoice relation respectively.

$\forall t_1 \in$ InvoiceDetail, $\exists t_2 \in$ Invoice: $t_1.$ID_Invoice = t_2.ID_Invoice

R9	Insert	Delete	Edit
InvoiceDetail	+	+	+(ID_Invoice)
Invoice	-	+	-

R10: In relationship InvoiceDetail and Product, attribute ID_Product of each line in the InvoiceDetail relation must correspond to the Product relation respectively.

$\forall t_1 \in$ InvoiceDetail, $\exists t_2 \in$ Product: $t_1.$ID_Product = t_2.ID_Product

R10	Insert	Delete	Edit
InvoiceDetail	+	-	+(ID_Product)
Product	-	+	-

R11: In relationship EnterCouponDetail and EnterCoupon, attribute ID_Enter of each line in the EnterCouponDetail relation must correspond to the EnterCoupon relation respectively.

$\forall t_1 \in$ EnterCouponDetail, $\exists t_2 \in$ EnterCoupon: $t_1.$ID_Enter = t_2.ID_Enter

R11	Insert	Delete	Edit
EnterCouponDetail	+	+	+(ID_Enter)
EnterCoupon	-	+	-

R12: In relationship EnterCouponDetail and Product, attribute ID_Product of each line in the InvoiceDetail relation must correspond to the Product relation respectively.

$\forall t_1 \in$ EnterCouponDetail, $\exists t_2 \in$ Product: $t_1.$ID_Product = t_2.ID_Product

R12	Insert	Delete	Edit
EnterCouponDetail	+	-	+(ID_Product)
Product	-	+	-

R13: In relationship Invoice, Product and InvoiceDetail, attribute TotalValue of one invoice must equal total value of attribute TotalValue of its details.

$$t.TotalValue = \sum_{\substack{x \in InvoiceDetail \\ \wedge x.ID\_Invoice = t.ID\_Invoice \\ \wedge x.ID\_Product = y.ID\_Product}} x.Quanity * y.Price\_Sale$$

$\forall$ t $\in$ Invoice $\wedge$ $\forall$ y $\in$ Product:

R13	Insert	Delete	Edit
Invoice	+	-	+(TotalValue)
InvoiceDetail	+	+	+(TotalValue, Quanity)
Product	-	+	+(Price_Sale)

R14: In relationship EnterCoupon, Product and EnterCouponDetail, attribute TotalValue of one enter coupon must equal total value of attribute TotalValue of its details.

$$t.TotalValue = \sum_{\substack{x \in EnterCouponDetail \\ \wedge x.ID\_Enter = t.ID\_Enter \\ \wedge x.ID\_Product = y.ID\_Product}} x.Quanity * y.Price\_Purchase$$

∀ t ∈ EnterCoupon ^ ∀ y ∈ Product:

R14	Insert	Delete	Edit
EnterCoupon	+	-	+(TotalValue)
EnterCouponDetail	+	+	+(TotalValue, Quanity)
Product	-	+	+(Price_Purchase)

6. Functional dependency and normalization

1. Customer(ID_Customer, NameCustomer, AddressCustomer, PhoneCustomer, BirthdayCustomer, GenderCustomer)

F: {ID_Customer → NameCustomer

ID_Customer → AddressCustomer

ID_Customer → PhoneCustomer

ID_Customer → BirthdayCustomer

ID_Customer → GenderCustomer}

So each functional dependency has ID_Customer as super key

=> In Boyce-Codd Normal Form

2. Employee(ID_Employee, ID_Position, NameEmployee, PhoneEmployee)

F: {ID_Employee → ID_Position

ID_Employee → NameEmployee

ID_Employee → PhoneEmployee}

So each functional dependency has ID_Employee as super key

=> In Boyce-Codd Normal Form

3. Position(ID_Position, Position, Permision)

F: {ID_Position → Position

ID_Position → Permision}

So each functional dependency has ID_Position as super key

=> In Boyce-Codd Normal Form

4. Product(ID_Product, ID_ProductClass, ID_Producer, NameProduct, Price_Purchase, Price_Sale, TotalQuanity, UpdateAt)
F: {ID_Product → ID_ProductClass

ID_Product → ID_Producer

ID_Product → NameProduct

ID_Product → Price_Purchase

ID_Product → Price_Sale

ID_Product → TotalQuanity

ID_Product → UpdateAt}

So each functional dependency has ID_Product as super key

=> In Boyce-Codd Normal Form

5. Producer(ID_Producer, NameProducer)
F: {ID_Producer → NameProducer}

So each functional dependency has ID_Producer as super key

=> In Boyce-Codd Normal Form

6. Vendor(ID_Vendor, NameVendor, AddressVendor, PhoneVendor)
F: {ID_Vendor → NameVendor

ID_Vendor → AddressVendor

ID_Vendor → PhoneVendor}

So each functional dependency has ID_Vendor as super key

=> In Boyce-Codd Normal Form

7. Invoice(ID_Invoice, ID_Customer, CreateAt, ID_Employee, TotalValue)
F: {ID_Invoice → ID_Customer

ID_Invoice → CreateAt

ID_Invoice → ID_Employee

ID_Invoice → TotalValue}

So each functional dependency has ID_Invoice as super key

=> In Boyce-Codd Normal Form

8. InvoiceDetail(ID_Invoice, ID_Product, Quanity, TotalValue)
F: {ID_Invoice, ID_Product → Quanity

ID_Invoice, ID_Product → TotalValue}

So each functional dependency has ID_Invoice, ID_Product as super key

=> In Boyce-Codd Normal Form

9. EnterCoupon(ID_Enter, ID_Employee, ID_Vendor, CreateAt, TotalValue)
F: {ID_Enter → ID_Employee

ID_Enter → ID_Vendor

ID_Enter → CreateAt

ID_Enter → TotalValue}

So each functional dependency has ID_Enter as super key

=> In Boyce-Codd Normal Form

10. EnterCouponDetail(ID_Enter, ID_Product, Quanity, TotalValue)
F: {ID_Enter, ID_Product → Quanity

ID_Enter, ID_Product → TotalValue}

So each functional dependency has ID_Enter, ID_Product as super key

=> In Boyce-Codd Normal Form

11. Inventory(Period, ID_Product, QuanityFirst, TotalQuanityPurchase, TotalQuanitySale, QuanityLast)
F: {Period, ID_Product → QuanityFirst

Period, ID_Product → TotalQuanityPurchase

Period, ID_Product → TotalQuanitySale

Period, ID_Product → QuanityLast}

So each functional dependency has Period, ID_Product as super key

=> In Boyce-Codd Normal Form

CHAPTER 3: REQUESTING INFORMATION FROM DATABASES

1.Sales operations

1.1 Create a list of invoice in month x year y. (To help managers supervise the sale)

SQL:

```
CREATE PROCEDURE ListOfInvoiceInMonthXYearX @Month int, @Year int
AS
SELECT *
FROM Invoice
WHERE MONTH(CreateAt)=@Month AND Year (CreateAt)=@Year

EXECUTE ListOfInvoiceInMonthXYearX @Month=1, @Year=2019
```

Result:

Results | Messages

	ID_Invoice	ID_Custom...	CreateAt	ID_Employ...	TotalVal...
1	HD01	KH02	2019-01-06	NVBH02	150000
2	HD02	KH04	2019-01-07	NVBH02	95000
3	HD03	KH05	2019-01-10	NVBH04	101000
4	HD04	KH01	2019-01-10	NVBH03	296000
5	HD05	KH02	2019-01-10	NVBH02	96000
6	HD06	KH03	2019-01-13	NVBH01	175000
7	HD07	KH07	2019-01-14	NVBH05	269000
8	HD08	KH08	2019-01-15	NVBH04	114000
9	HD09	KH01	2019-01-15	NVBH02	75200
10	HD10	KH04	2019-01-18	NVBH01	183400
11	HD11	KH05	2019-01-19	NVBH03	142500
12	HD12	KH08	2019-01-19	NVBH04	150000
13	HD13	KH09	2019-01-21	NVBH05	175000
14	HD14	KH10	2019-01-22	NVBH01	152100
15	HD15	KH04	2019-01-23	NVBH03	214600

Qu... | LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (51) | ANOCompany | 00:00:00 | 20 rows

1.2 Calculate revenue and profits of months of year x. In which month do you have the highest revenue? (To help managers manage business strategies and import goods)

SQL:

```
CREATE PROCEDURE CalculateRevenueAndProfitOfMonthsInYearX @Year int
AS
SELECT Month(inv.CreateAt) as Month, Sum(invdl.TotalValue) as Revenue,
Sum(invdl.TotalValue - invdl.Quanity*p.Price_Purchase) as Profit
From InvoiceDetail as invdl inner join Product as P on P.ID_Product= invdl.ID_Product
                    inner join Invoice as inv on invdl.ID_Invoice=inv.ID_Invoice
WHERE YEAR(inv.CreateAt)=@Year
GROUP BY Month(inv.CreateAt)
ORDER BY Profit Desc

EXECUTE CalculateRevenueAndProfitOfMonthsInYearX @Year =2019
```

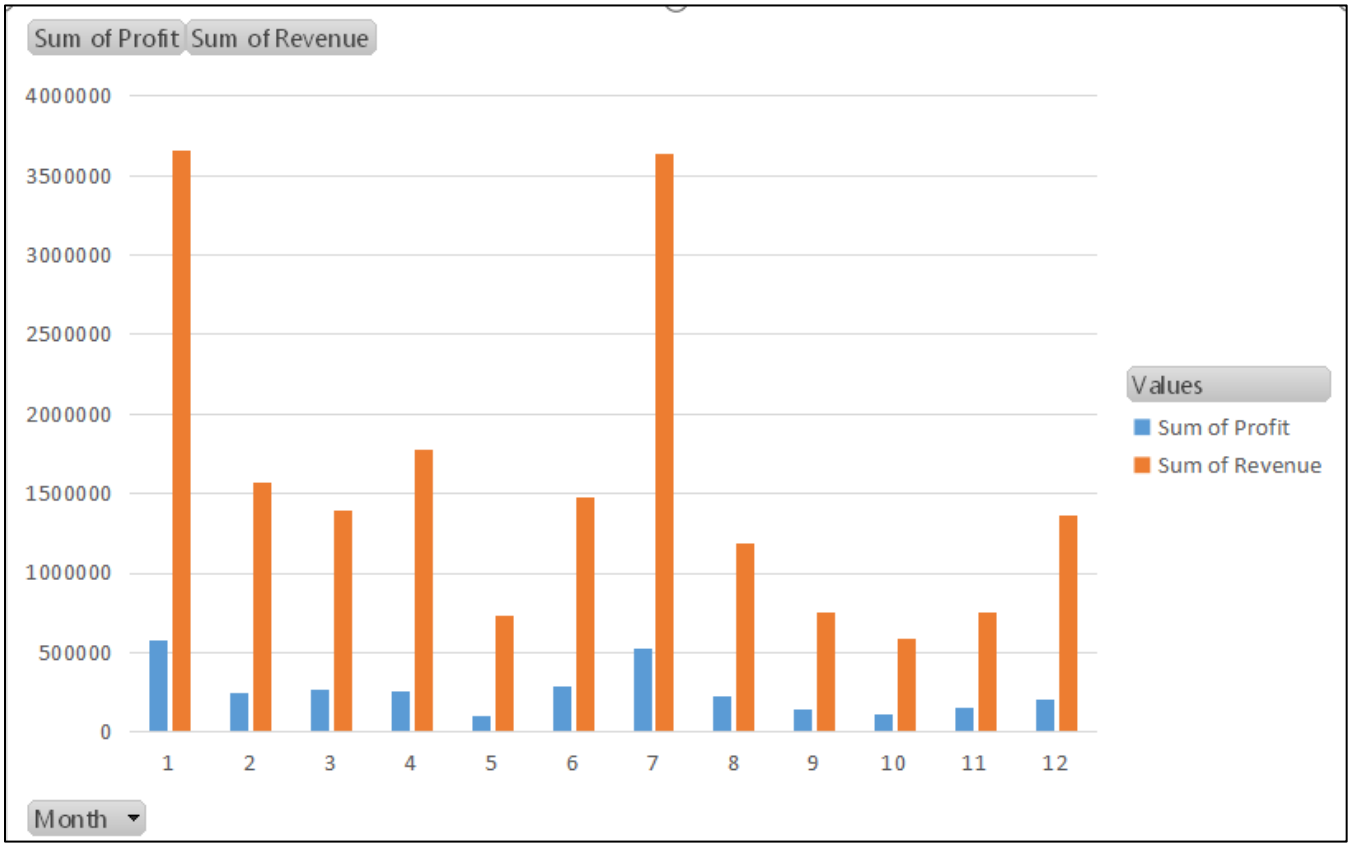
Result:

Results | Messages

	Month	Revenue	Profit
1	1	3654400	573700
2	7	3632800	520000
3	6	1472600	285000
4	3	1390500	265000
5	4	1776200	260000
6	2	1572200	245000
7	8	1181600	220000
8	12	1358600	200000
9	11	756300	150000
10	9	754000	145000
11	10	586700	115000
12	5	730400	100000

LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (51) | ANOCompany | 00:00:00 | 12 rows

Report:



From this chart, we can see that the revenue of January and July in 2019 is the highest, and the profit in these 2 months is also the highest. It is speculated that because in January people will shop a lot for the Tet holiday, while July is in the middle of the summer, the weather is quite hot and children are on vacation, playing, learning more. So the demand for milk will increase in those 2 months. Besides that, sales in September, October, and November are relatively low in 2019. From here, the company will actively import goods to meet market demand.

1.3 Create a list of products sold, total sold, revenue and profit in month x year y. (To help managers supervise inventory)

SQL:

```
CREATE PROCEDURE ListOfProductsSoldTotalSoldRenueveAndProfitInMonthXYearY
@Month int, @Year int
AS
SELECT invdl.ID_Product, Sum(invdl.Quanity) as TotalSold, Sum(invdl.TotalValue) as
        Renueve, Sum(invdl.Quanity*(p.Price_Sale-p.Price_Purchase)) as Profit
FROM InvoiceDetail as invdl inner join Product as P on invdl.ID_Product=P.ID_Product
                inner join Invoice as inv on invdl.ID_Invoice=inv.ID_Invoice
WHERE MONTH(inv.CreateAt)=@Month and YEAR(inv.CreateAt)=@Year
GROUP BY invdl.ID_Product
ORDER BY TotalSold DESC

EXECUTE ListOfProductsSoldTotalSoldRenueveAndProfitInMonthXYearY @Month=1,
@Year=2019
```

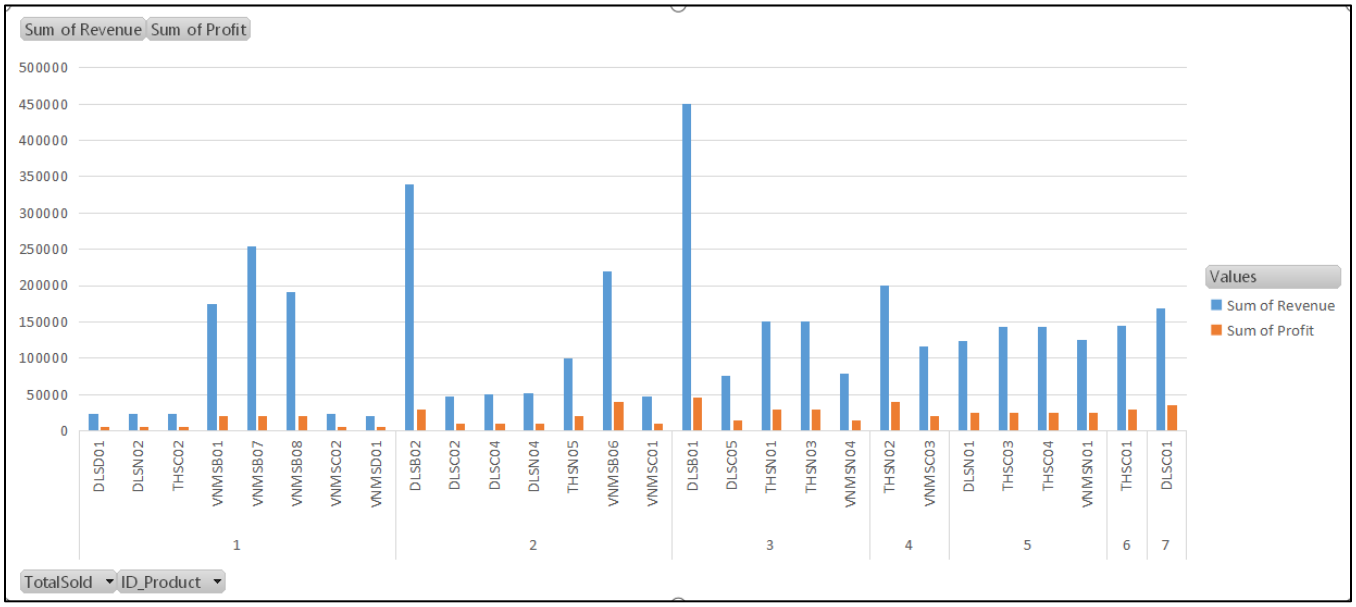
Result:

Results | Messages

	ID_Product	TotalS...	Reven...	Profit
1	DLSC01	7	168000	35000
2	THSC01	6	144000	30000
3	THSC03	5	142500	25000
4	THSC04	5	142500	25000
5	DLSN01	5	123000	25000
6	VNMSN01	5	125000	25000
7	VNMSC03	4	115600	20000
8	THSN02	4	200000	40000
9	THSN03	3	150000	30000
10	THSN01	3	150000	30000
11	DLSC05	3	75000	15000
12	DLSB01	3	450000	45000
13	VNMSN04	3	78000	15000
14	VNMSB06	2	219000	40000
15	VNMSC01	2	47200	10000

Q | LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (52) | ANOCompany | 00:00:00 | 28 rows

Report:



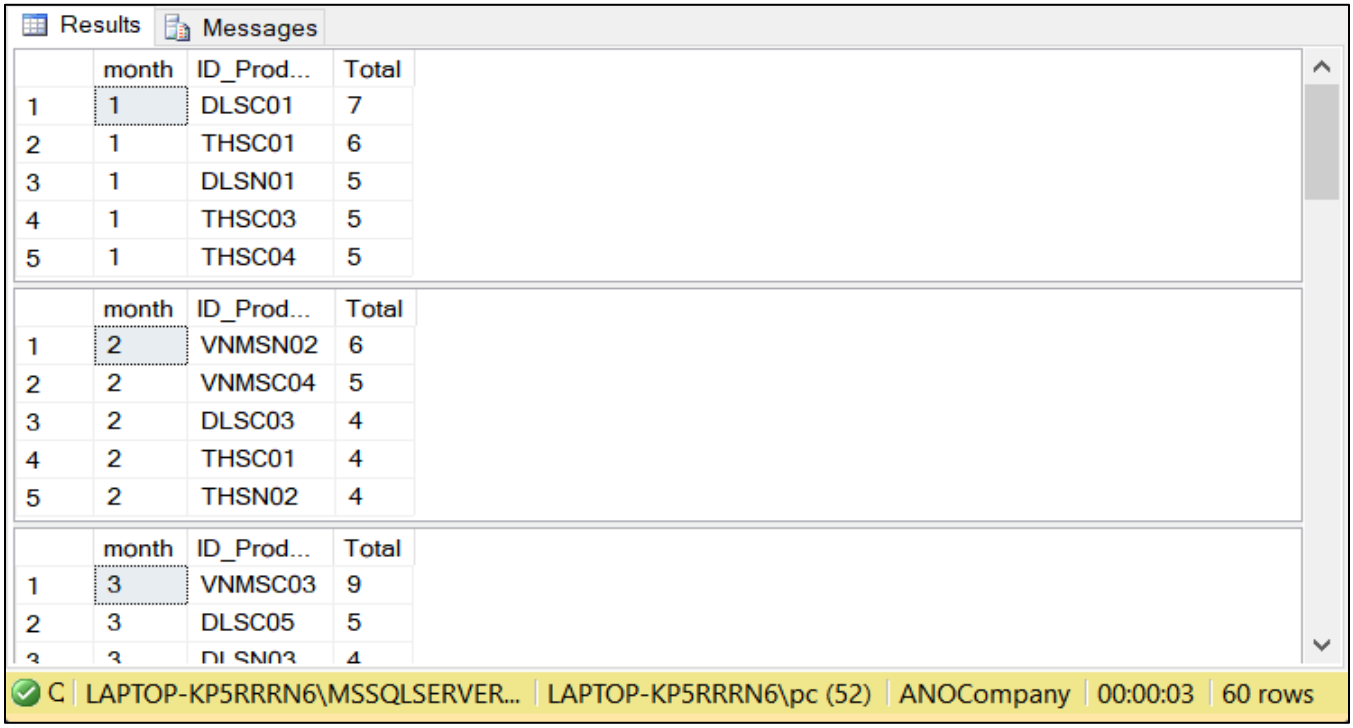
ANO Company has 43 types of dairy products, but only 28 types are sold in January 2019, of which the most popular are yogurt and liquid milk products. Although powdered milk products are sold in smaller quantities than yogurt and liquid milk, they bring a big revenue and profit because the price of milk powder products is much higher than yogurt and liquid milk. From this, realize that not all products of the company have market share, the company needs to change the product list to limit inventory for too long. The company should focus on importing yogurt and liquid milk products, restricting the import of less favored condensed and powdered milk products.

1.4 Top 5 the most sold products (in the month and in the year x). (To help managers manage business strategies).

SQL: In each month of year x

```
CREATE PROCEDURE Top5TheMostSoldProductInEachMonthOfYearX @Year int
AS
BEGIN
	DECLARE @Month int=0
		WHILE @Month<12
			BEGIN
				SET @Month=@Month+1;
				SELECT TOP(5) Month(inv.CreateAt) as Month, invdl.ID_Product,

			SUM(invdl.Quanity) as Total
				FROM InvoiceDetail as invdl inner join Product as p on

			invdl.Id_Product=p.Id_Product inner join Invoice as inv on

			invdl.ID_Invoice=invdl.ID_Invoice
				WHERE		MONTH(inv.CreateAt)=@Month		and
YEAR(INV.CreateAt)=@Year
				GROUP BY MONTH(inv.CreateAt), invdl.ID_Product
				ORDER BY SUM(invdl.Quanity) DESC
			END
END

EXECUTE Top5TheMostSoldProductInEachMonthOfYearX @Year =2019
```

Result:



Results | Messages

	month	ID_Prod...	Total
1	1	DLSC01	7
2	1	THSC01	6
3	1	DLSN01	5
4	1	THSC03	5
5	1	THSC04	5

	month	ID_Prod...	Total
1	2	VNMSN02	6
2	2	VNMSC04	5
3	2	DLSC03	4
4	2	THSC01	4
5	2	THSN02	4

	month	ID_Prod...	Total
1	3	VNMSC03	9
2	3	DLSC05	5
3	3	DLSN03	4

C | LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (52) | ANOCompany | 00:00:03 | 60 rows

SQL: In year

```
CREATE PROCEDURE Top5TheMostSoldProductsInYearX @Year INT
AS
SELECT TOP(5) invdl.ID_Product, Sum(invdl.Quanity) as TotalSold
FROM InvoiceDetail as invdl inner join Product as p on invdl.ID_Product = p.ID_Product
                    inner join Invoice as inv on invdl.ID_Invoice = inv.ID_Invoice
WHERE YEAR(inv.CreateAt) = @Year
GROUP BY invdl.ID_Product
ORDER BY TotalSold Desc

EXEC Top5TheMostSoldProductsInYearX @Year=2019
```

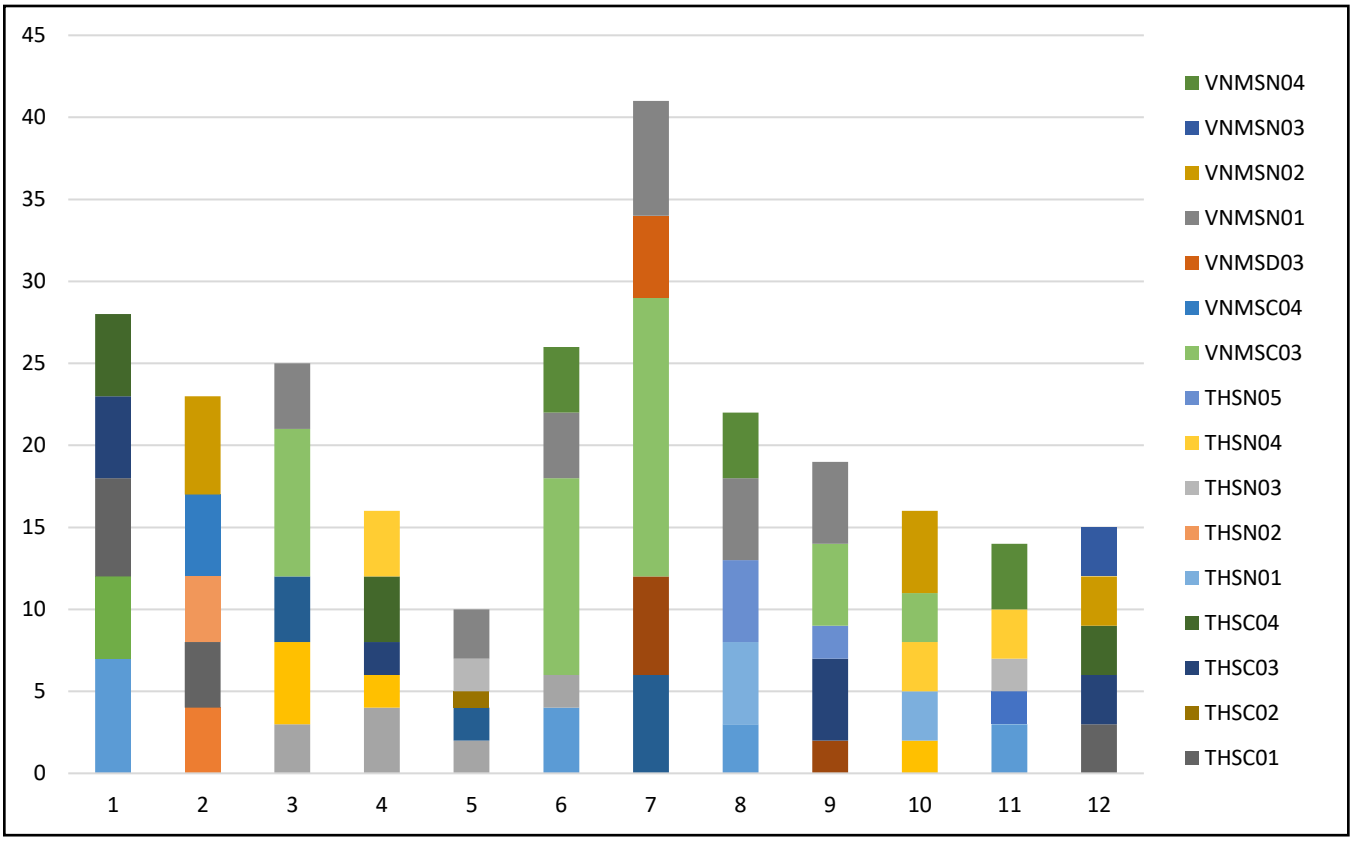
Result:

Results | Messages

	ID_Product	TotalS...
1	VNMSC03	52
2	VNMSN01	35
3	THSC03	24
4	VNMSN04	22
5	THSN01	18

LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (57) | ANOCompany | 00:00:00 | 5 rows

Report:



1.5 Create a list of how much revenue and profit each customer bringing in each month in year x?
(To help managers know well each customer)

SQL:

```
CREATE PROCEDURE ListOfHowMuchRevenueAndProfitEachCustomerBringingInEachMonthOfYearX @Year int
AS
SELECT MONTH(inv.CreateAt) AS Month, inv.ID_Customer, SUM(inv.TotalValue) as Revenue,
       SUM(invdl.Quanity*(p.price_Sale - p.Price_Purchase)) as Profit
FROM Customer as cus inner join Invoice as inv on cus.ID_Customer = inv.ID_Customer
        inner join InvoiceDetail as invdl on inv.ID_Invoice = invdl.ID_Invoice
        inner join Product as p on invdl.ID_Product = p.ID_Product
WHERE Year(inv.CreateAt) = @Year
GROUP BY inv.ID_Customer,MONTH(inv.CreateAt)
ORDER BY inv.ID_Customer

EXEC ListOfHowMuchRevenueAndProfitEachCustomerBringingInEachMonthOfYearX @Year=2019
```

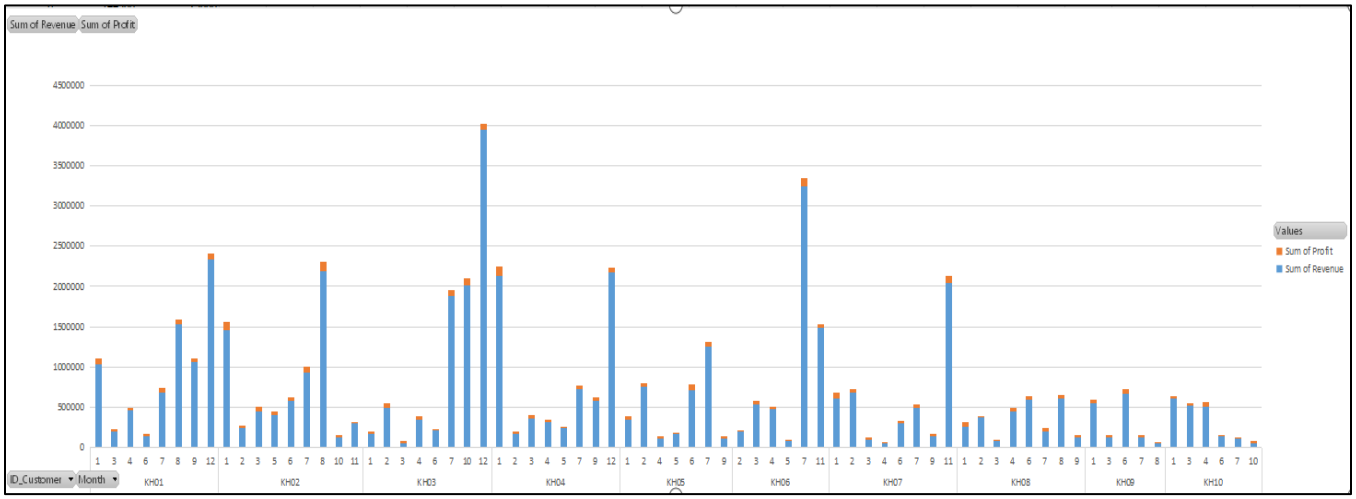
Result:

Results | Messages

	Month	ID_Custom...	Revenue	Profit
1	1	KH01	1038400	75000
2	3	KH01	206400	20000
3	4	KH01	462600	30000
4	6	KH01	144500	25000
5	7	KH01	683200	55000
6	8	KH01	1532400	60000
7	9	KH01	1068600	35000
8	12	KH01	2341200	65000
9	1	KH02	1451200	115000
10	2	KH02	248000	25000
11	3	KH02	441200	60000
12	5	KH02	397200	50000
13	6	KH02	584700	35000
14	7	KH02	923600	85000
15	8	KH02	2192000	110000

Q | LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (56) | ANOCompany | 00:00:00 | 73 rows

Report:



Based on this chart, we see the revenue and profit that each customer contributes to the company every month in 2019. Customer with code KH03 is the most profitable and revenue for the company, then the customers KH02, KH01, KH06. From here, the company will have preferential policies for these customers in the next business period. Besides, customers KH10, KH09, KH08 bring the lowest profit and revenue, the company will research the causes (geographical distance, product demand, ...) to take measures to stimulate demand fit.

1.6 Revenue of each producer by month in year x. (To help managers manage import goods)

SQL:

```
CREATE PROCEDURE RevenueOfEachProducerByMonthInYearX @year int
AS
SELECT pr.NameProducer, Sum(inv.TotalValue) as Total, MONTH(CreateAt) as Month
FROM InvoiceDetail as invdl inner join Product as p on invdl.ID_Product = p.ID_Product
                    inner join Category as ca on p.ID_ProductClass = ca.ID_ProductClass
                inner join Producer as pr on p.ID_Producer = pr.ID_Producer
                        inner join Invoice as inv on invdl.ID_Invoice = inv.ID_Invoice
WHERE YEAR(inv.CreateAt) = @year
GROUP BY  pr.NameProducer, MONTH(inv.CreateAt)

EXEC RevenueOfEachProducerByMonthInYearX @Year=2019
```

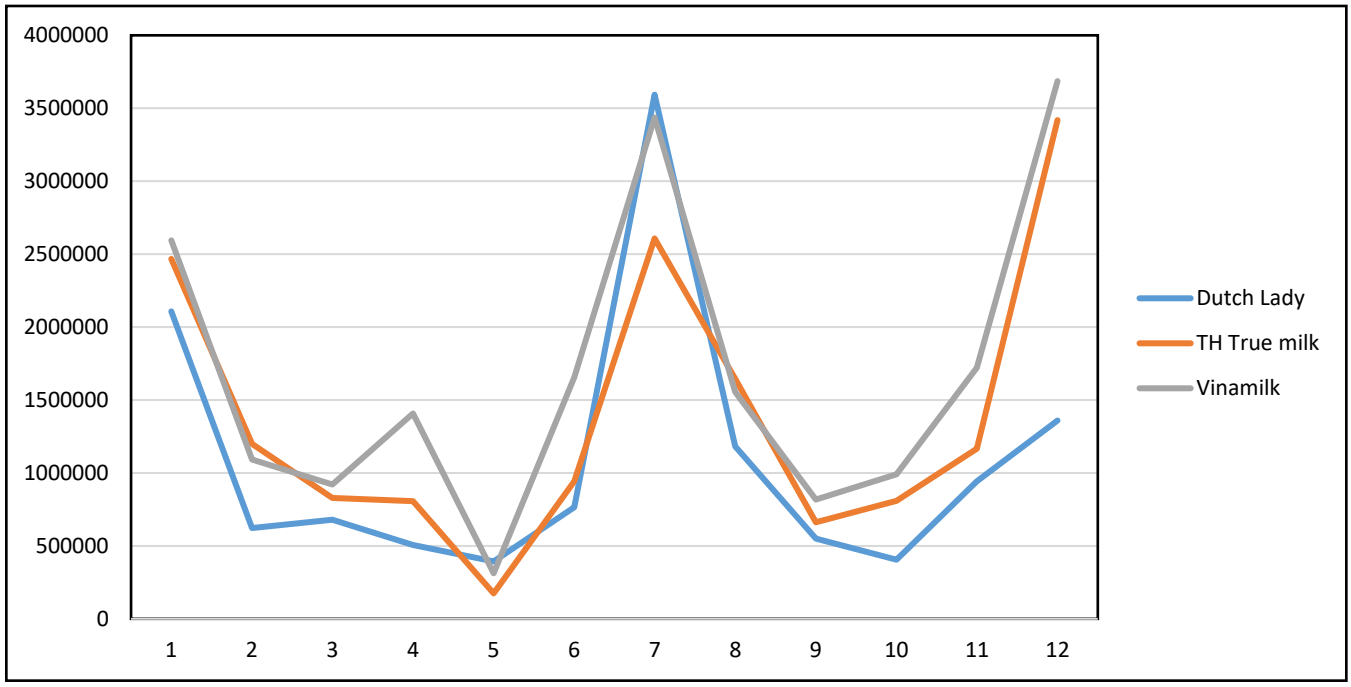
Result:

Results | Messages

	NameProdu...	Total	Month
1	Dutch Lady	2106400	1
2	TH True milk	2468000	1
3	Vinamilk	2594900	1
4	Dutch Lady	621200	2
5	TH True milk	1198700	2
6	Vinamilk	1090700	2
7	Dutch Lady	680200	3
8	TH True milk	828600	3
9	Vinamilk	921300	3
10	Dutch Lady	506800	4
11	TH True milk	806000	4
12	Vinamilk	1408000	4
13	Dutch Lady	393200	5
14	TH True milk	174000	5
15	Vinamilk	311800	5

LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (52) | ANOCompany | 00:00:00 | 36 rows

Report:



In general, in 2019, Vinamilk's milk products are the most bought and Dutch Lady is the least popular. Almost every month, Vinamilk's milk products account for a lot of sales, so it is necessary to focus on importing Vinamilk's products and cut down Dutch Lady.

1.7 Revenue of each category by month in year x. (To help managers know well each customer)

SQL:

```
CREATE PROCEDURE RevenueOfEachCategoryByMonthInYearX @year int
AS
SELECT Ca.Category, Sum(inv.TotalValue) as Total, MONTH(CreateAt) as Month
FROM InvoiceDetail as invdl inner join Product as p on invdl.ID_Product = p.ID_Product
                        inner join Category as ca on p.ID_ProductClass = ca.ID_ProductClass
                    inner join Producer as pr on p.ID_Producer = pr.ID_Producer
                            inner join Invoice as inv on invdl.ID_Invoice = inv.ID_Invoice
WHERE YEAR(inv.CreateAt) = @year
GROUP BY  Ca.Category, MONTH(inv.CreateAt)

EXEC RevenueOfEachCategoryByMonthInYearX @Year=2019
```

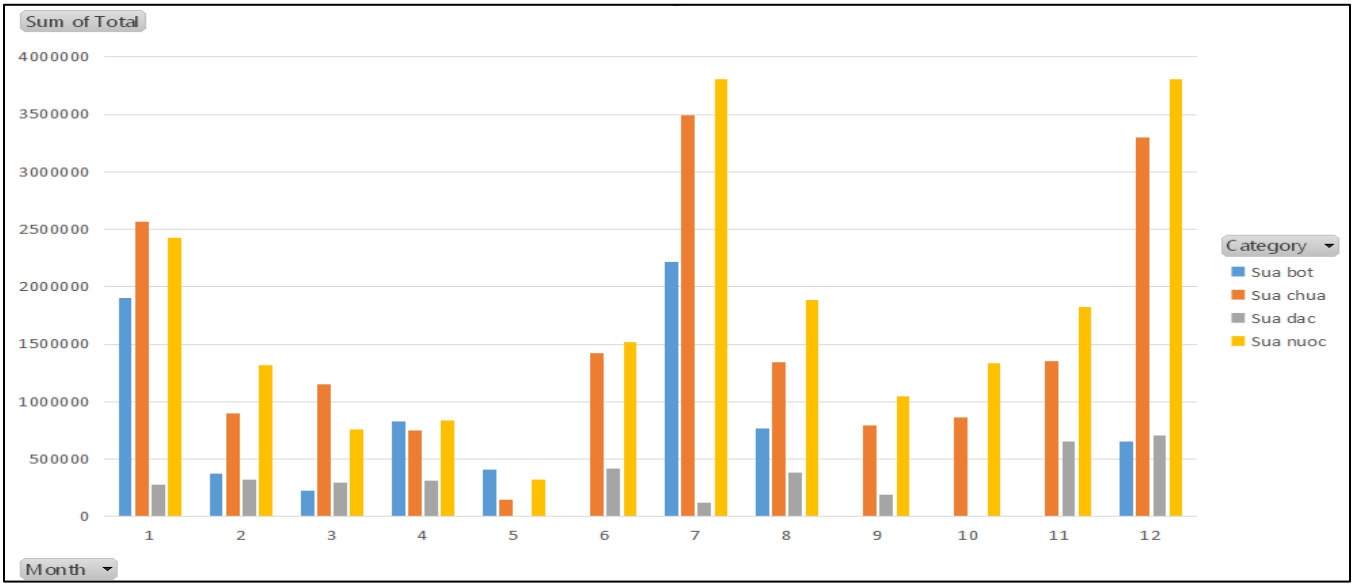
Result:

Results | Messages

	Category	Total	Month
1	Sua bot	1901200	1
2	Sua chua	2566500	1
3	Sua dac	279400	1
4	Sua nuoc	2422200	1
5	Sua bot	371500	2
6	Sua chua	898700	2
7	Sua dac	321200	2
8	Sua nuoc	1319200	2
9	Sua bot	225600	3
10	Sua chua	1147100	3
11	Sua dac	296600	3

Quer... | LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (56) | ANOCompany | 00:00:00 | 42 rows

Report:



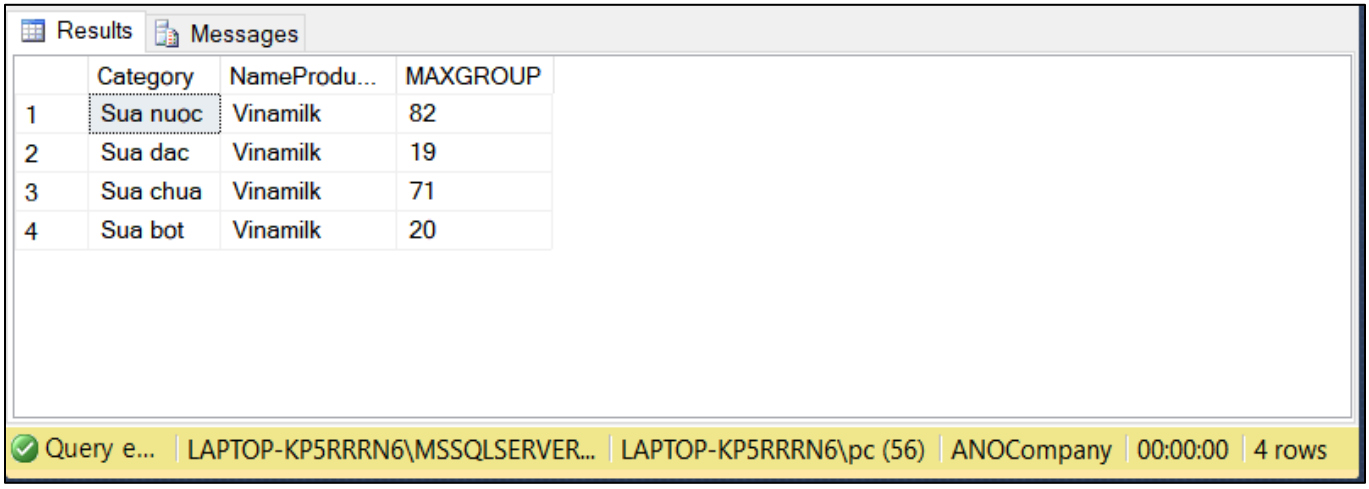
Revenue in 2019 from liquid milk is the highest and powdered milk is the lowest. Because the consumption demand for liquid milk products is much higher than that of powdered milk products, the company needs to shift its import of powdered milk products to liquid milk products, then yogurt. Should focus on a variety of liquid milk and yogurt products to meet market needs.

1.8 For the same category (such as powdered milk), which producer's products are the best selling in year x? (To help managers catch customer's interested and to support import decision making)

SQL:

```
CREATE PROCEDURE WhichProducerHasTheMostSoldProDuctInTheSameCategoryInYearX
@Year int
AS
SELECT  t.Category, t.NameProducer, a.MAXGROUP
FROM(SELECT ca.Category, pr.NameProducer, SUM(invdl.Quanity) AS Quantity1
    FROM InvoiceDetail as invdl inner join Product as p on invdl.ID_Product = p.ID_Product
                        inner join Category as ca  on p.ID_ProductClass=ca.ID_ProductClass
                    inner join Producer as pr on p.ID_Producer = pr.ID_Producer
                                inner join Invoice as inv on invdl.ID_Invoice = inv.ID_Invoice
     WHERE YEAR(inv.CreateAt) = @Year
     GROUP BY Ca.Category, Pr.NameProducer) as t
          INNER JOIN
     (SELECT  a.Category, MAX(a.Quantity1) MAXGROUP
      FROM(SELECT ca.Category, Pr.NameProducer, SUM(invdl.Quanity) AS Quantity1
          FROM InvoiceDetail as invdl inner join Product as p on invdl.ID_Product=p.ID_Product
                        inner join Category as ca on p.ID_ProductClass=ca.ID_ProductClass
                    inner join Producer as pr on p.ID_Producer = pr.ID_Producer
                                inner join Invoice as inv on invdl.ID_Invoice = inv.ID_Invoice
          WHERE YEAR(inv.CreateAt) = @Year
          GROUP BY ca.Category, pr.NameProducer) as a
      GROUP BY a.Category) a ON t.Category = a.Category AND t.Quantity1 = a.MAXGROUP

EXEC WhichProducerHasTheMostSoldProDuctInTheSameCategoryInYearX @Year=2019
```

Result:

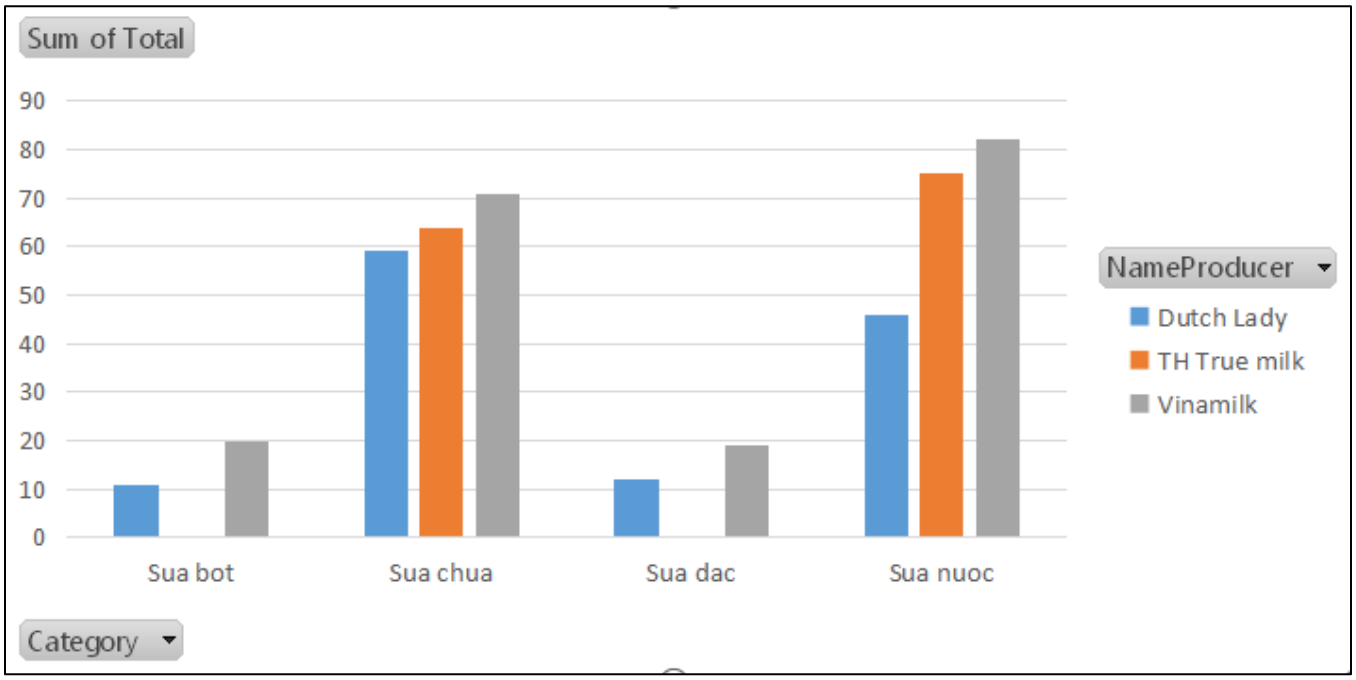


Results | Messages

	Category	NameProdu...	MAXGROUP
1	Sua nuoc	Vinamilk	82
2	Sua dac	Vinamilk	19
3	Sua chua	Vinamilk	71
4	Sua bot	Vinamilk	20

Query e... | LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (56) | ANOCompany | 00:00:00 | 4 rows

Report:



From this chart, we see that with the same category of dairy product, Vinamilk producer is always more favored by consumers. Therefore, the company should focus on diversifying Vinamilk's products rather than Dutch Lady and TH True Milk to meet customers' needs.

1.9 With the x-type category, which ones do customers buy with x. (To know buying habits of customers)

SQL:

```
CREATE PROCEDURE ListProductWithX @ID_ProductX nchar(10)
AS
SELECT  inv.ID_Product, p.NameProduct, COUNT(*) as Appearance
FROM InvoiceDetail inv inner join Product p on inv.ID_Product = p.ID_Product
WHERE ID_Invoice in (SELECT ID_Invoice
                     FROM InvoiceDetail
                     WHERE InvoiceDetail.ID_Product = @ID_ProductX)
  and inv.ID_Product <> @ID_ProductX
GROUP BY inv.ID_Product ,p.NameProduct
ORDER BY COUNT(*) DESC

EXEC ListProductWithX @ID_ProductX='VNMSC01'
```

Result:

Results | Messages

	ID_Product	NameProduct	Appearance
1	VNMSC03	Sua chua an nha dam	3
2	VNMSD01	Sua dac PHUONG NAM 380g	1
3	DLSC01	Sua chua an co duong	1
4	DLSD01	Sua dac nguyen kem 380g	1
5	DLSN02	Sua tiet trung khong duong	1
6	THSC03	Sua chua an nha dam	1
7	THSN01	Sua hat va gac 180mlx4	1
8	THSN02	Sua hat va nghe 180mlx4	1
9	THSN03	Sua hat hanh nhan 180mlx4	1
10	VNMSC02	Sua chua an khong duong	1

Qu... | LAPTOP-KP5RRRN6\MSSQLSERVER... | LAPTOP-KP5RRRN6\pc (53) | ANOCompany | 00:00:00 | 10 rows

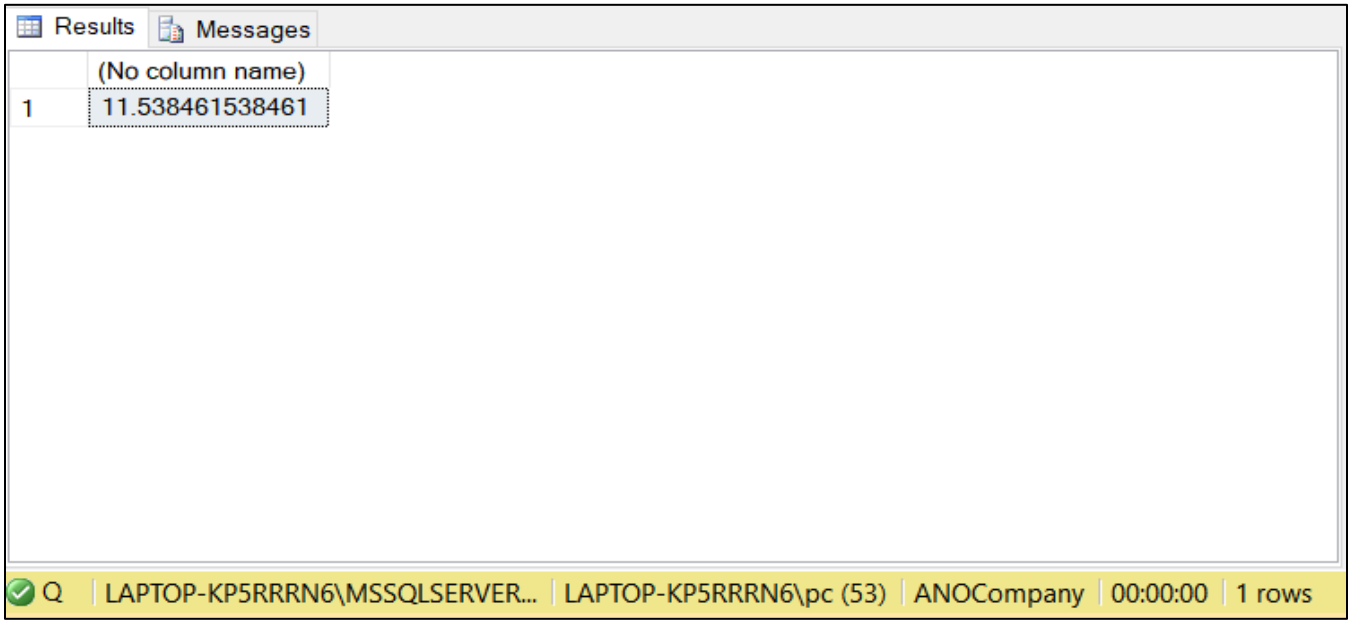
1.10 Check the reliability of this statement: If customers buy the type category x and the y type category at the same time, they will usually buy the z-category (evaluated based on revenue). (To know about buying habits of customers)

SQL: Test with random category x = 'SN', y = 'SC' and z = 'SB'

```
CREATE PROCEDURE CheckCategoryXYZ @X NCHAR(10), @Y NCHAR(10), @Z
NCHAR(10)
AS
BEGIN
SELECT (SELECT COUNT(P.ID_Invoice)
      FROM(SELECT INV.ID_Invoice, ISNULL((SELECT SUM(invdl.Quanity*p.Price_Sale)
                                          FROM InvoiceDetail as invdl inner join Product as p
on
                                    invdl.ID_Product = p.ID_Product
                                          WHERE invdl.ID_Invoice = inv.ID_Invoice
                                GROUP BY P.ID_ProductClass
                                HAVING P.ID_ProductClass = @X), 0) AS X,
                                     ISNULL((SELECT SUM(invdl.Quanity*p.Price_Sale)
                                          FROM InvoiceDetail as invdl inner join Product as p
on
                                    invdl.ID_Product = p.ID_Product
                                          WHERE invdl.ID_Invoice = inv.ID_Invoice
                                          GROUP BY P.ID_ProductClass
                                     HAVING P.ID_ProductClass = @Y), 0) AS Y,
                                     ISNULL((SELECT SUM(invdl.Quanity*P.Price_Sale)
                                          FROM InvoiceDetail as invdl inner join Product as p on
                                   invdl.ID_Product = p.ID_Product
                                WHERE invdl.ID_Invoice = inv.ID_Invoice
                                GROUP BY P.ID_ProductClass
                                HAVING P.ID_ProductClass =  @Z), 0) AS Z
         FROM Invoice AS INV) AS P
        WHERE P.X>0 AND P.Y>0 AND P.Z>0) * 100.0/
          (SELECT COUNT(P.ID_Invoice)
         FROM(SELECT INV.ID_Invoice, ISNULL((SELECT
SUM(invdl.Quanity*p.Price_Sale)
                                FROM InvoiceDetail as invdl inner join Product as
                                   p on invdl.ID_Product = p.ID_Product
                                WHERE invdl.ID_Invoice = inv.ID_Invoice
                                GROUP BY P.ID_ProductClass
```

```
                                HAVING P.ID_ProductClass = @X), 0) AS X,
                                        ISNULL((SELECT SUM(invdl.Quanity*p.Price_Sale)
                                FROM InvoiceDetail as invdl inner join Product as
                                    p on invdl.ID_Product = p.ID_Product
                                WHERE invdl.ID_Invoice = inv.ID_Invoice
                                GROUP BY P.ID_ProductClass
                                HAVING P.ID_ProductClass = @Y), 0) AS Y,
                                        ISNULL((SELECT SUM(invdl.Quanity*P.Price_Sale)
                                FROM InvoiceDetail as invdl inner join Product as
                                    p on invdl.ID_Product = p.ID_Product
                                WHERE invdl.ID_Invoice = inv.ID_Invoice
                                GROUP BY P.ID_ProductClass
                                HAVING P.ID_ProductClass =  @Z), 1) AS Z
            FROM Invoice AS INV)AS P
        WHERE P.X>0 AND P.Y>0)
END;

EXEC CheckCategoryXYZ @X='SN', @Y='SC', @Z='SB'
```

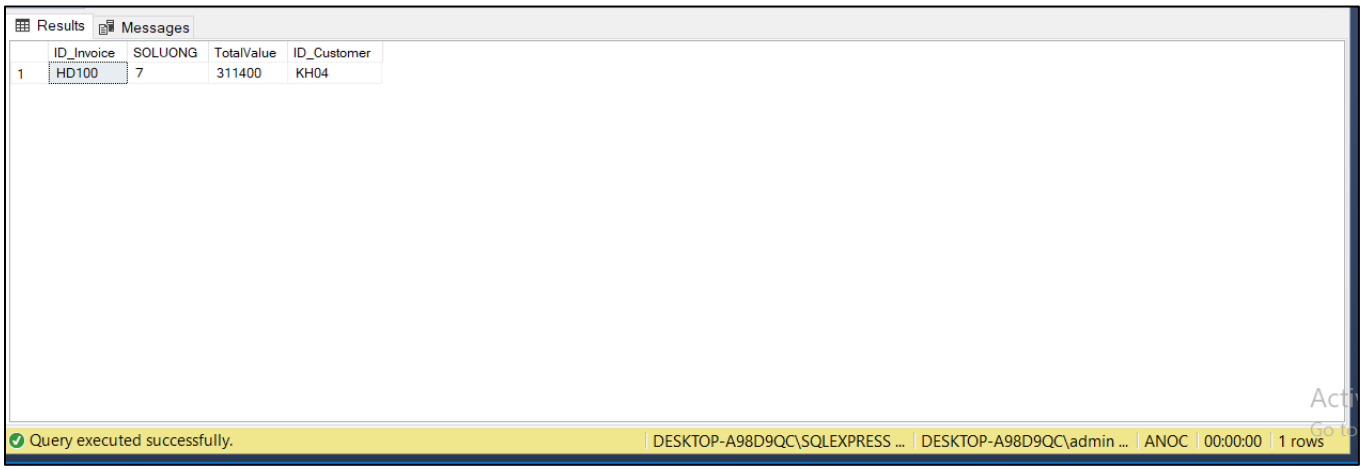
Result: The statement: “If customers buy the type category ‘SN’ and the ‘SC’ type category at the same time, they will usually buy the ‘SB’ - category “ has 11.54% reliability. That means 88,46% risk of mistake



1.11 Find out which invoice have most items in month x, year y.

```
CREATE PROCEDURE QuantityProduct @MONTH DATETIME, @YEAR DATETIME
AS
 SELECT  TOP(1)  Invoice.ID_Invoice,  COUNT(InvoiceDetail.ID_Product)  AS  SOLUONG,
Invoice.TotalValue, Invoice.ID_Customer
 FROM InvoiceDetail INNER JOIN Invoice ON Invoice.ID_Invoice=InvoiceDetail.ID_Invoice
 WHERE MONTH(Invoice.CreateAt)=12 AND YEAR(Invoice.CreateAt)=2019
 GROUP BY Invoice.ID_Invoice, Invoice.TotalValue, Invoice.ID_Customer
```

```
EXEC QuantityProduct @MONTH=12, @YEAR=2019
```

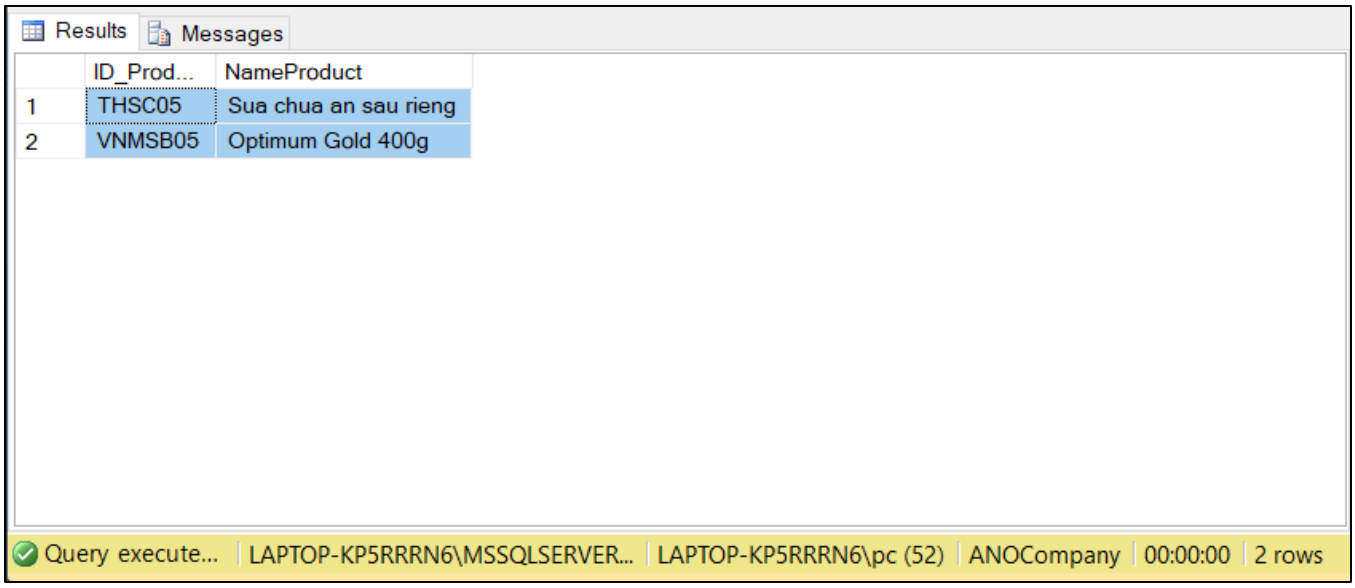


1.12 Create a list of items which are not purchased in year x (to help the managers adjust their busniness strategy or make some marketing programs with those items).

SQL:

```
CREATE PROCEDURE ProductNotSaleInYearX @Year datetime
AS
SELECT Product.ID_Product, Product.NameProduct
FROM Product
WHERE Product.ID_Product NOT IN
(
SELECT InvoiceDetail.ID_Product
FROM InvoiceDetail INNER JOIN Invoice ON Invoice.ID_Invoice=InvoiceDetail.ID_Invoice
WHERE YEAR(Invoice.CreateAt)=@Year
)
 EXEC ProductNotSaleInYearX @YEAR=2019
```

Result:



Report:

ID_Product	NameProduct
THSC05	Sua chua an sau rieng
VNMSB05	Optimum Gold 400g

1.13 Make a list of invoices issued from day x to day y.

SQL:

```
CREATE PROCEDURE Invoice_InvoiceFromDateXtoDateY @DateX datetime,@DateY datetime
AS
SELECT *
FROM Invoice
WHERE (Invoice.CreateAt) between @DateX and @DateY

EXEC Invoice_InvoiceFromDateXtoDateY @DateX ='2010-01-02',@DateY='2019-02-09'
```

Result:

	ID_Invoice	ID_Customer	CreateAt	ID_Employee	TotalValue
1	HD01	KH02	2019-01-06 00:00:00.000	NVBH02	150000
2	HD02	KH04	2019-01-07 00:00:00.000	NVBH02	95000
3	HD03	KH05	2019-01-10 00:00:00.000	NVBH04	101000
4	HD04	KH01	2019-01-10 00:00:00.000	NVBH03	296000
5	HD05	KH02	2019-01-10 00:00:00.000	NVBH02	96000
6	HD06	KH03	2019-01-13 00:00:00.000	NVBH01	175000
7	HD07	KH07	2019-01-14 00:00:00.000	NVBH05	269000
8	HD08	KH08	2019-01-15 00:00:00.000	NVBH04	114000
9	HD09	KH01	2019-01-15 00:00:00.000	NVBH02	75200
10	HD10	KH04	2019-01-18 00:00:00.000	NVBH01	183400
11	HD11	KH05	2019-01-19 00:00:00.000	NVBH03	142500
12	HD12	KH08	2019-01-19 00:00:00.000	NVBH04	150000
13	HD13	KH09	2019-01-21 00:00:00.000	NVBH05	175000
14	HD14	KH10	2019-01-22 00:00:00.000	NVBH01	152100
15	HD15	KH04	2019-01-23 00:00:00.000	NVBH03	214600
16	HD16	KH02	2019-01-24 00:00:00.000	NVBH03	404600
17	HD17	KH03	2019-02-06 00:00:00.000	NVBH02	50000
18	HD18	KH05	2019-02-06 00:00:00.000	NVBH04	250000
19	HD19	KH06	2019-02-06 00:00:00.000	NVBH05	100000
20	HD49	KH04	2019-01-06 00:00:00.000	NVBH03	388000
21	HD50	KH02	2019-01-06 00:00:00.000	NVBH05	300000

Query executed succ... | ANBINH0777\SQLEXPRESS (12.0... | ANBINH0777\MYCOM (58) | ANOCompany | 00:00:00 | 23 rows

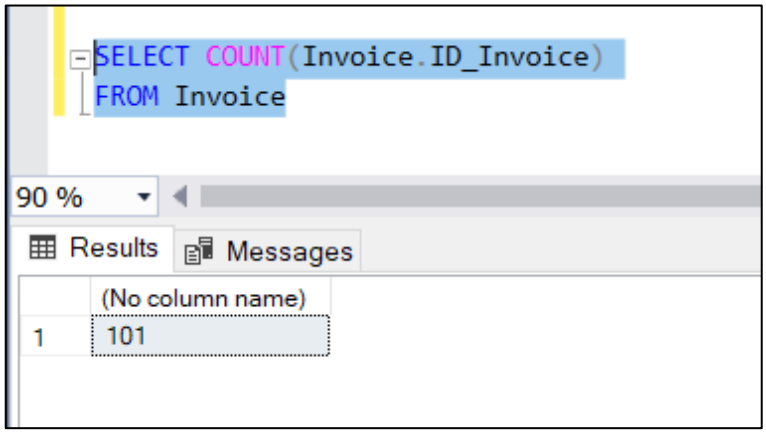
1.14 Perform queries insert, delete, edit data

- Insert: The managers can use this procedure to insert one or many invoices in their database.

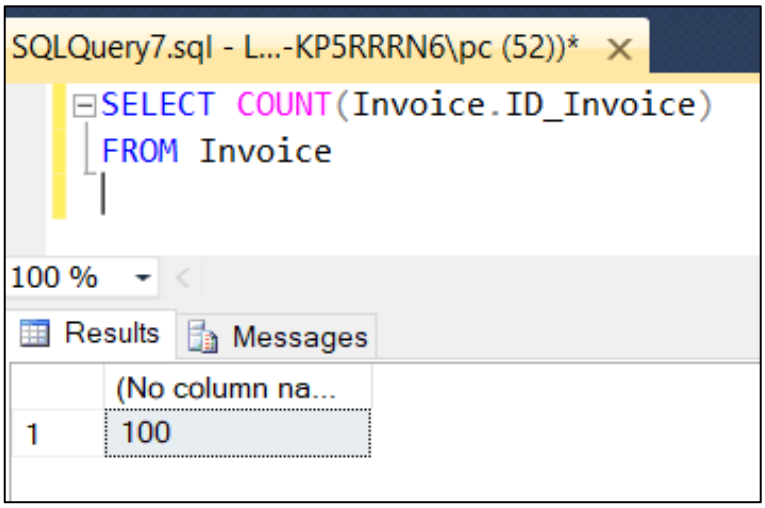
```
CREATE PROCEDURE INSERT_INVOICE @idinvoice nchar(10)=null, @idcustomer nchar(10)=null, @at datetime=null, @idemployee nchar(10)=null, @value float=null
AS
BEGIN
    INSERT INTO Invoice(ID_Invoice, ID_Customer, CreateAt, ID_Employee, TotalValue)
    VALUES(@idinvoice, @idcustomer, @at, @idemployee, @value)
END
GO
```

EX: They insert 1 more invoice in the Invoice table.

Before: There are 100 rows in Invoice that means there are 100 invoices.

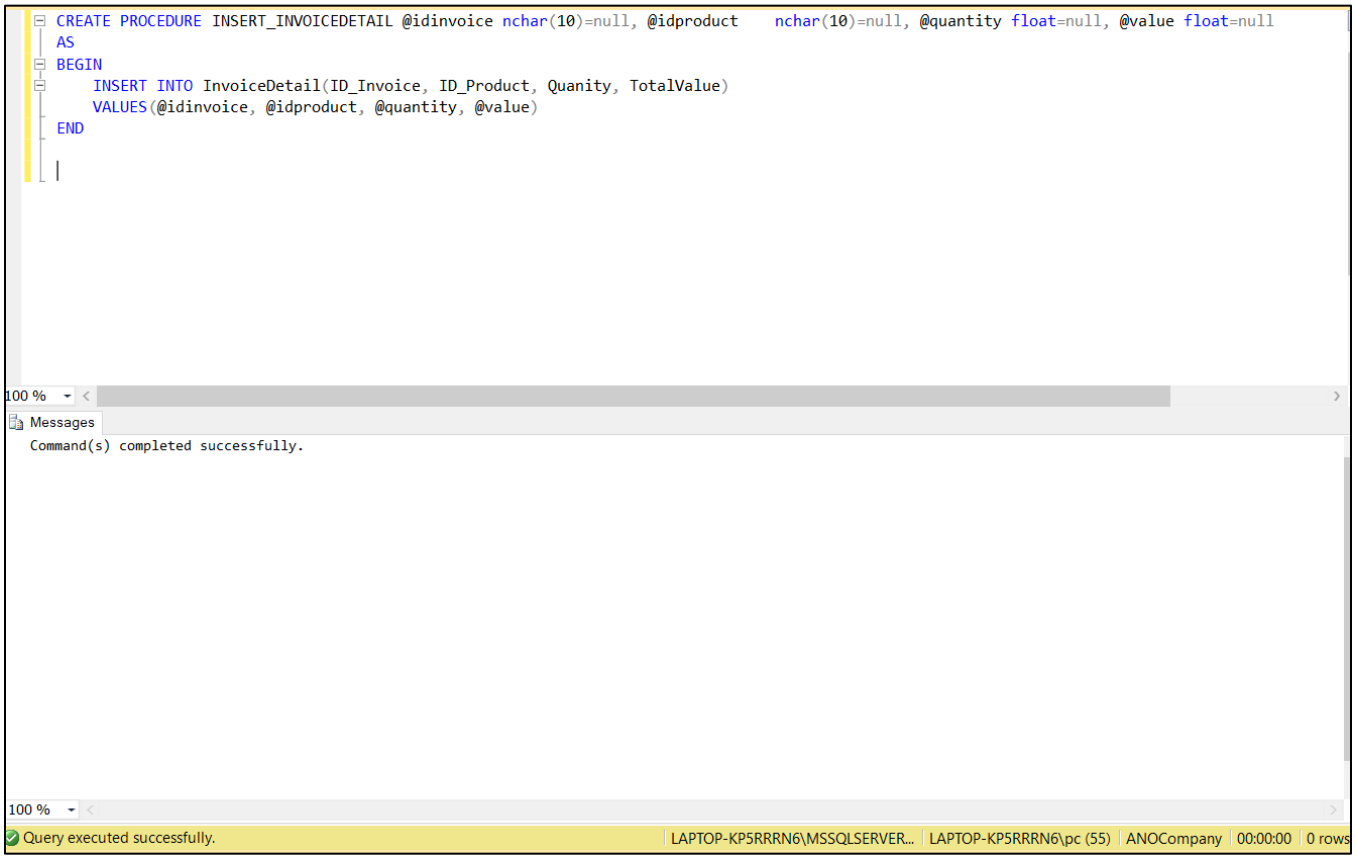


After: We've just added a new invoice (HD101) so that the total rows now are 101 rows.



But remember that when you add a new invoice in Invoice table, you also need to add more information in InvoiceDetail table too.

```
CREATE PROCEDURE INSERT_INVOICEDETAIL @idinvoice nchar(10)=null, @idproduct nchar(10)=null, @quantity float=null, @value float=null
AS
BEGIN
    INSERT INTO InvoiceDetail(ID_Invoice, ID_Product, Quanity, TotalValue)
    VALUES(@idinvoice, @idproduct, @quantity, @value)
END
GO
```



- Update: The managers can use the procedure to change any attributes in one invoice if they find some changes.

```
CREATE PROCEDURE UPDATE_INVOICE @idinvoice nchar(10)=null, @idcustomer nchar(10)=null, @at datetime=null, @idemployee nchar(10)=null, @value float=null
AS
BEGIN
    UPDATE Invoice
        SET ID_Customer=@idcustomer, CreateAt=@at, ID_Employee=@idemployee, TotalValue=@value
    WHERE ID_Invoice=@idinvoice
END
GO
```

EX: They need to change ID_Customer in HD01 when they find a fault.

Before:

	ID_Invoice	ID_Customer	CreateAt	ID_Employee	TotalValue
1	HD01	KH02	2019-01-06 00:00:00.000	NVBH02	150000
2	HD02	KH04	2019-01-07 00:00:00.000	NVBH02	95000
3	HD03	KH05	2019-01-10 00:00:00.000	NVBH04	101000
4	HD04	KH01	2019-01-10 00:00:00.000	NVBH03	296000
5	HD05	KH02	2019-01-10 00:00:00.000	NVBH02	96000

After:

	ID_Invoice	ID_Customer	CreateAt	ID_Employee	TotalValue
1	HD01	KH01	2019-01-06 00:00:00.000	NVBH02	150000
2	HD02	KH04	2019-01-07 00:00:00.000	NVBH02	95000
3	HD03	KH05	2019-01-10 00:00:00.000	NVBH04	101000
4	HD04	KH01	2019-01-10 00:00:00.000	NVBH03	296000
5	HD05	KH02	2019-01-10 00:00:00.000	NVBH02	96000

- Delete: The managers donnot need to delete any invoices because they are historical data and they need them to make some exact decisions.

1. Warehouse management operations

2.1 Create a list of imported items in month x year y (to help the warehouse department check stock status)

SQL:

```
CREATE PROCEDURE ProductImport @MONTH DATETIME
AS
SELECT
EnterCouponDetail.ID_Product,Category.Category, Product.NameProduct
,EnterCouponDetail.Quanity, (EnterCouponDetail.TotalValue) as TotalPrice
FROM
((EnterCouponDetail INNER JOIN EnterCoupon ON
EnterCouponDetail.ID_Enter=EnterCoupon.ID_Enter)
INNER JOIN Product ON Product.ID_Product=EnterCouponDetail.ID_Product)
INNER JOIN Category ON Category.ID_ProductClass=Product.ID_ProductClass
WHERE MONTH(EnterCoupon.CreateAt)=@MONTH

EXEC ProductImport @MONTH=3
```

Result:

Results | Messages

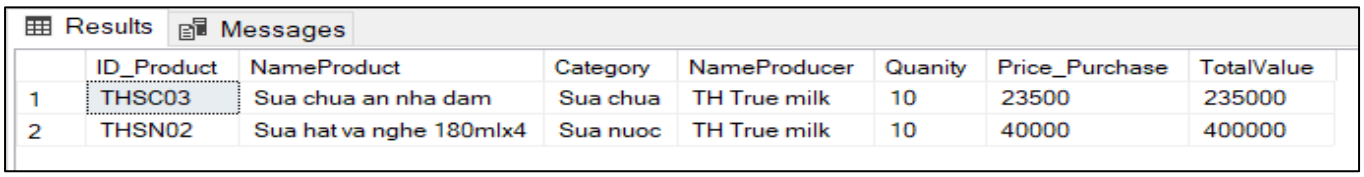
	ID_Product	Category	NameProduct	Quanity	TotalPrice
1	DLSC03	Sua chua	Sua chua an nha dam	10	200000
2	THSC01	Sua chua	Sua chua an co duong	10	190000
3	THSC04	Sua chua	Sua chua an dua	10	235000
4	THSN01	Sua nuoc	Sua hat va gac 180mlx4	10	400000
5	VNMSC04	Sua chua	Sua chua an it duong	10	190000
6	VNMSN01	Sua nuoc	Sua tuoi co duong 180mlx4	10	200000
7	VNMSN02	Sua nuoc	Sua tuoi khong duong 180mlx4	10	200000
8	VNMSN03	Sua nuoc	Sua tuoi huong dau 180mlx4	10	210000

2.2 Create a list of the transaction with y vendor in month x year y

SQL:

```
CREATE PROCEDURE VendorValueMonth @VENDOR NCHAR(10), @MONTH DATETIME
AS
SELECT
EnterCouponDetail.ID_Product, Product.NameProduct, Category.Category, producer.NameProducer, EnterCouponDetail.Quanity, Product.Price_Purchase, (ProductDetail.Price_Purchase*EnterCouponDetail.Quanity) AS TotalValue
FROM
(((EnterCouponDetail INNER JOIN EnterCoupon ON EnterCouponDetail.ID_Enter=EnterCoupon.ID_Enter) INNER JOIN Product ON Product.ID_Product=EnterCouponDetail.ID_Product) INNER JOIN category ON product.ID_ProductClass=category.ID_ProductClass) INNER JOIN producer ON Producer.ID_Producer=Product.ID_Producer
WHERE
  EnterCoupon.ID_Vendor=@VENDOR AND MONTH(EnterCoupon.CreateAt)=@MONTH

  EXEC VendorValueMonth @VENDOR='C02', @MONTH=2;
```

Result:



Results | Messages

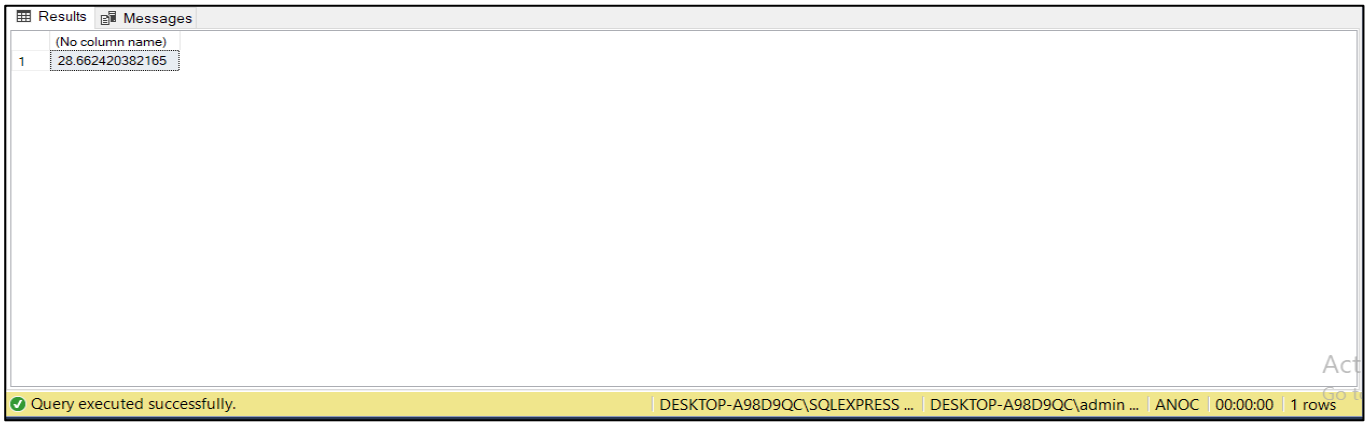
	ID_Product	NameProduct	Category	NameProducer	Quanity	Price_Purchase	TotalValue
1	THSC03	Sua chua an nha dam	Sua chua	TH True milk	10	23500	235000
2	THSN02	Sua hat va nghe 180mlx4	Sua nuoc	TH True milk	10	40000	400000

2.3 Test the reliability of the statement: "Customers tend to buy all items of the same supplier" (help the managers draw a suitable plan to get higher profit)

SQL:

```
SELECT (SELECT COUNT (q.ID_Invoice)
      FROM  (SELECT InvoiceDetail.ID_Invoice, Producer.NameProducer
            FROM InvoiceDetail INNER JOIN Product ON
InvoiceDetail.ID_Product=Product.ID_Product
                  INNER JOIN Producer ON Product.ID_Producer=Producer.ID_Producer
            GROUP BY InvoiceDetail.ID_Invoice, Producer.NameProducer
            HAVING Count(ID_Invoice)>1)AS q) * 100.0/
           (SELECT  COUNT (q.ID_Invoice)
                FROM  (SELECT InvoiceDetail.ID_Invoice, Producer.NameProducer
                FROM InvoiceDetail INNER JOIN Product ON
InvoiceDetail.ID_Product=Product.ID_Product
                      INNER JOIN Producer ON
Product.ID_Producer=Producer.ID_Producer
                GROUP BY InvoiceDetail.ID_Invoice, Producer.NameProducer)AS q)
```

Result:



Results | Messages

	(No column name)
1	28.662420382165

Query executed successfully. | DESKTOP-A98D9QC\SQLEXPRESS ... | DESKTOP-A98D9QC\admin ... | ANOC | 00:00:00 | 1 rows

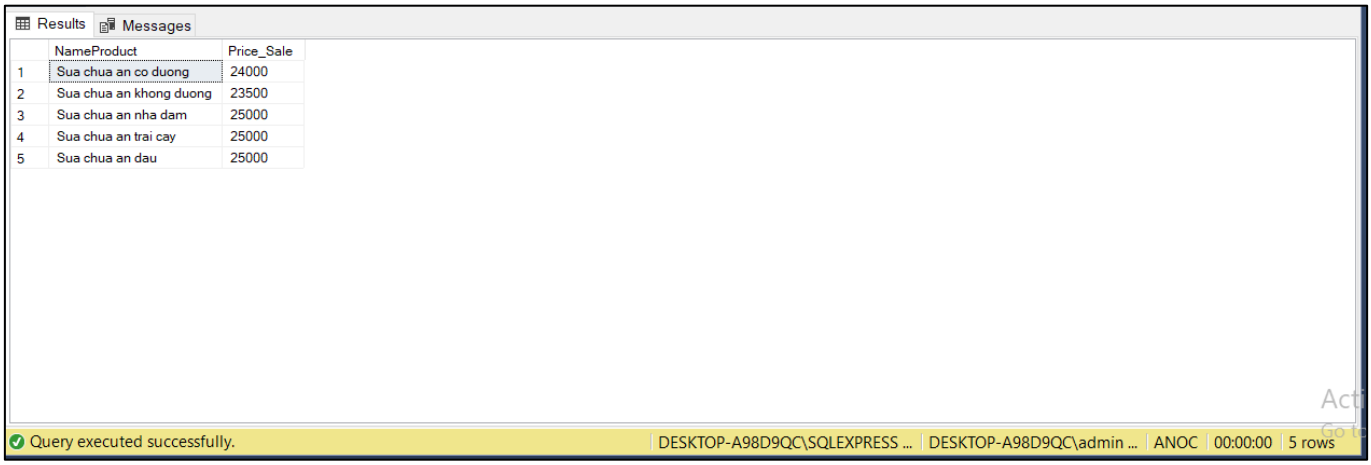
"The statement "Customers are tend to buy all items of the same supplier" gets 28,66% reliability, which means it gets 71.34% risk of mistake. According to this result, the managers cannot believe in that statement thoroughly but it also help them to make some plans to achieve higher profit, such as the arrangement in their stores or promotion for the same brand."

2.4 Create a list of items from x (dong) to y (dong), followed by z producer (to help the managers in the sales)

SQL:

```
CREATE PROCEDURE ProductFollowByPrice @PRICE1 float, @PRICE2 float, @CATEGORY nvarchar(100), @IDPRODUCER nvarchar(10)
AS
SELECT Product.NameProduct, Product.Price_Sale
FROM Product INNER JOIN Category ON Product.ID_ProductClass=Category.ID_ProductClass
  WHERE (Product.Price_Sale BETWEEN @PRICE1 AND @PRICE2) AND Category.Category=@CATEGORY AND Product.ID_Producer=@IDPRODUCER

  EXEC ProductFollowByPrice @PRICE1=20000, @PRICE2=50000, @CATEGORY='Sua chua', @IDPRODUCER='DL';
```

Result:



2.5 Perform queries insert, delete, edit data

- Insert:

```
CREATE PROCEDURE INSERT_COUPON @identer nchar(10)=null, @idemployee nchar(10)=null, @idvendor nchar(10)=null, @at datetime, @value float=null
AS
BEGIN
   INSERT INTO EnterCoupon(ID_Enter, ID_Employee, ID_Vendor, CreateAt, TotalValue)
   VALUES(@identer, @idemployee, @idvendor, @at, @value)
END
GO
```

EX:

Before: There are 25 Coupons.

After: We've just added a new coupons so there are 26 Coupons now.

But remember that when you add a new coupon in EnterCoupon table, you also need to add more information in EnterCouponDetail table too. (the same case like INSERT_INVOICEDETAIL procedure)

- Update: The managers can use the procedure to change any attributes in one coupon if they find some changes.

```
CREATE PROCEDURE UPDATE_COUPON @identer nchar(10)=null, @idemployee nchar(10)=null, @idvendor nchar(10)=null, @at datetime, @value float=null
AS
BEGIN
    UPDATE EnterCoupon
    SET ID_Employee=@idemployee, ID_Vendor=@idvendor, CreateAt=@at, TotalValue=@value
    WHERE ID_Enter=@identer
END
GO
```

EX: We want to change ID_Vendor in PN01.

Before:

	ID_Enter	ID_Employee	ID_Vendor	CreateAt	TotalValue
1	PN01	NVTK01	C01	2019-01-01 00:00:00.000	1388000
2	PN02	NVTK02	C02	2019-01-01 00:00:00.000	3085000
3	PN03	NVTK01	C03	2019-01-01 00:00:00.000	1411000

After:

	ID_Enter	ID_Employee	ID_Vendor	CreateAt	TotalValue
1	PN01	NVTK01	C03	2019-01-01 00:00:00.000	1388000
2	PN02	NVTK02	C02	2019-01-01 00:00:00.000	3085000
3	PN03	NVTK01	C03	2019-01-01 00:00:00.000	1411000

- Delete: The managers donnot need to delete any coupons because they are historical data and they need them to make some exact decisions.

3.Financial-accounting operations

3.1. In order for managers to have an overview of the company's financial situation in the months of the year, the Accounting-Finance Department needs to have a price report in each period. In order to statistic the degree of inventory price fluctuation in that year. From there, plan a warehousing strategy to match the company's financial and financial resources.

Requirements: The table of statistics includes the warehousing prices for the months of that year.

SQL

```
CREATE PROCEDURE InventoryPriceEachMonthInYearX  @Year int
AS
SELECT                                  Month(Inventory.Period)                  AS
Month_,SUM(Inventory.QuanityLast*Product.Price_Purchase)AS InventoryPrice
FROM Inventory inner join Product on Inventory.ID_Product = Product.ID_Product
WHERE YEAR(Inventory.Period)=@Year
GROUP BY  Month(Inventory.Period)
```

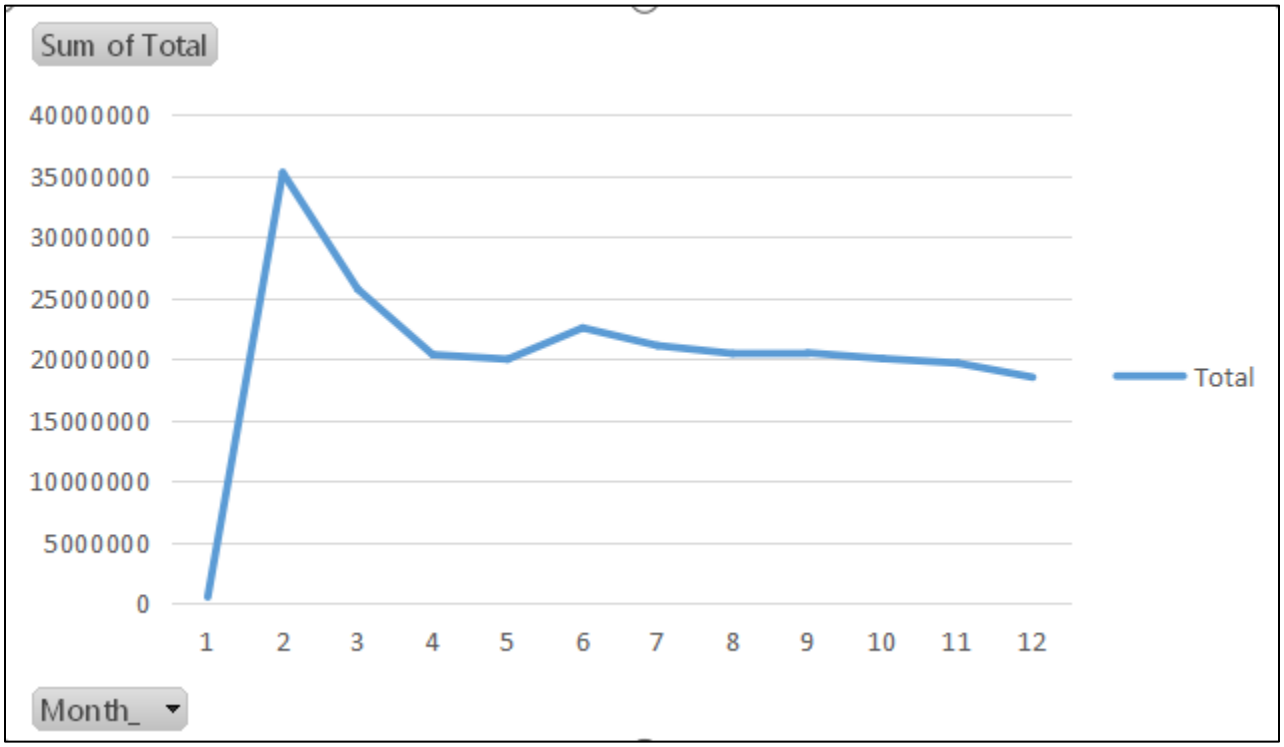
```
EXEC WarehousingCostOfYearX  @Year=2019
```

Result:

Results | Messages

	Mont...	InventoryPrice
1	1	540000
2	2	35265200
3	3	25736200
4	4	20371200
5	5	19994100
6	6	22563500
7	7	21110700
8	8	20464600
9	9	20520200
10	10	20048500
11	11	19681200
12	12	18522600

Report:



Based on the chart, we see in 2019 the highest inventory value in February and began to decline gradually in the following others. The reason February has a high inventory value while January is very low is probably because people are busy shopping for Tet in January. After all, they worry that February will be difficult to buy goods because of rising prices and scarcity. rare goods due to Tet.

3.2. An enterprise only exists when its goods and products are exchanged and traded in the market, ANO Company is also among them. For the exchange to be effective, the company needs to carefully monitor its sales. In addition, the company should combine with market research and conduct reasonable price adjustment strategies. The average value of all invoices sold in each month of the year is a prerequisite for implementing this strategy.

Requirements: The table includes the average of invoices for each month

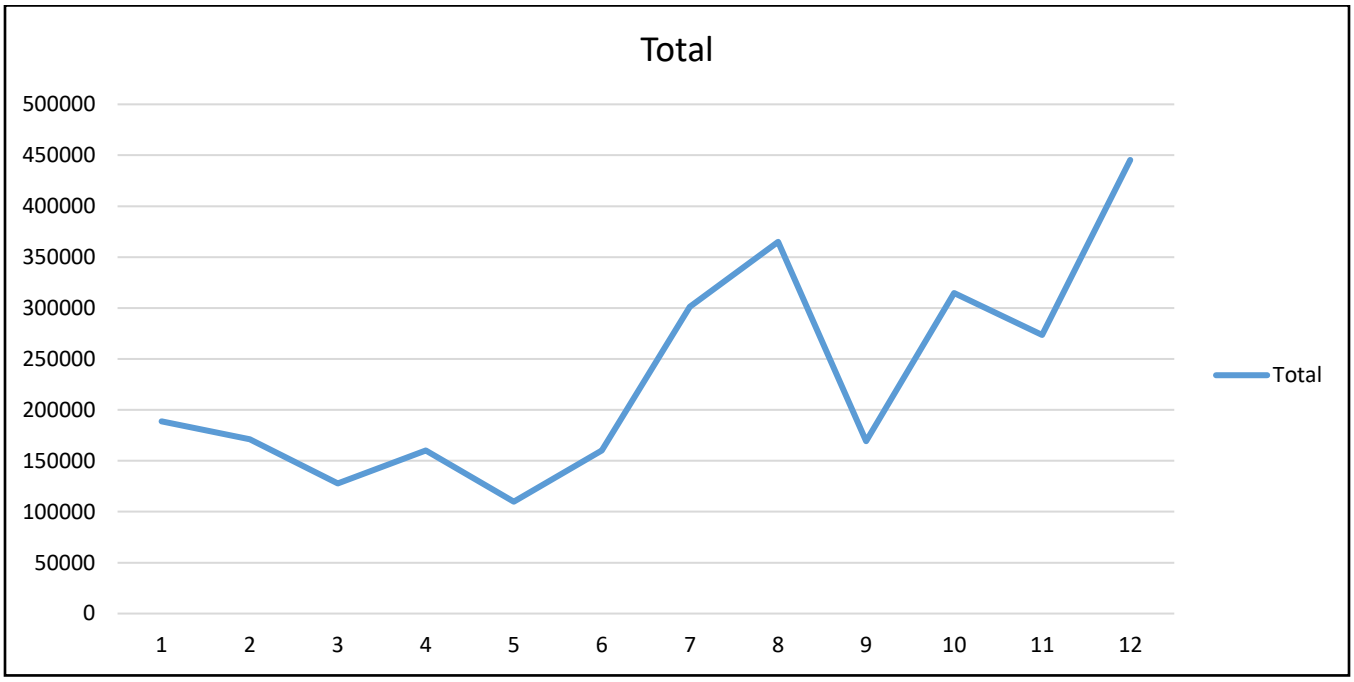
SQL:

```
CREATE PROCEDURE AverageInvoiceOfYearX  @Year int
AS
SELECT  Month(Invoice.CreateAt)  AS  Month_,ROUND(AVG(Invoice.TotalValue),2)AS AverageInvoice
FROM  InvoiceDetail inner join Invoice on InvoiceDetail.ID_Invoice=Invoice.ID_Invoice
WHERE YEAR(Invoice.CreateAt)=@Year
GROUP BY  Month(Invoice.CreateAt)

EXEC AverageInvoiceOfYearX @Year=2019
```

Result:

	Month_	AverageInvoice
1	1	188665.79
2	2	171211.76
3	3	127900
4	4	160047.06
5	5	109875
6	6	160019.05
7	7	301178.13
8	8	364991.67
9	9	169108.33
10	10	314614.29
11	11	273471.43
12	12	445421.05

Report:



In general, the average value of all invoices sold in each month of 2019 fluctuates continuously. Value increased during the period from May to August then dropped sharply in September, but did not decrease continuously in the following months.

4. Relations with suppliers and customers

4.1 To create close relationships with customers, the store has special promotions every month for all customers whose birthdays are in the month. Please print this customer list (including all customer information) with a birthday for that month. *Updated (5% off) on total bill for that month. In order to implement the strategy "Happy Customers' Birthday*".

SQL:

```
CREATE PROCEDURE DiscountWithBirthdayInMonthX @Month nchar(10)
AS
SELECT Customer.*, Invoice.TotalValue*(1-0.05) as Total_AfterDiscount
FROM Invoice inner join Customer on Invoice.ID_Customer=Customer.ID_Customer
WHERE MONTH(Customer.BirthdayCustomer)=@Month and
Month(Invoice.CreateAt)=@Month

EXEC DiscountWithBirthdayInMonthX  @Month=3
```

Result:

	ID_Customer	NameCustomer	AddressCustomer	PhoneCustomer	BirthdayCustomer	GenderCustomer	Total_AfterDiscount
1	KH02	Tran Ngoc Han	23/5 Nguyen Trai, Q5, TpHCM	908256478	1974-03-04	female	102410
2	KH09	Le Ha Vinh	873 Le Hong Phong, Q5, TpHCM	8654763	1979-03-09	male	118750
3	KH02	Tran Ngoc Han	23/5 Nguyen Trai, Q5, TpHCM	908256478	1974-03-04	female	214320

Query executed successfully. | ANBINH0777\SQLEXPRESS (12.0... | ANBINH0777\MYCOM (58) | ANOCompany | 00:00:00 | 3 rows

Report:

ID_Customer	NameCustomer	AddressCustomer	PhoneCustomer	BirthdayCustomer	GenderCustomer	Total_AfterDisocunt
KH02	Tran Ngoc Han	23/5 Nguyen Trai, Q5, TpHCM	908256478	3/4/1974	female	102410
KH09	Le Ha Vinh	873 Le Hong Phong, Q5, TpHCM	8654763	3/9/1979	male	118750
KH02	Tran Ngoc Han	23/5 Nguyen Trai, Q5, TpHCM	908256478	3/4/1974	female	214320

Based on the list of customers whose birthdays are in April, the company will apply special preferential policies for these customers when purchasing in April.

4.2 Through Customer Relationship Management (CRM) to find, attract, new customers, maintain existing partners, entice old customers back, reduce marketing costs and expand customer service. . Besides, maintaining an intimate relationship with potential customers is also essential. The company needs to identify potential customers and their incentives by giving Discount coupons, giving priority to trial of high-end products, receiving holiday offers, ..

Requirements: Among the 5 customers with the highest sales, find customers with the highest number of purchases each month of the year.

SQL

```
CREATE PROCEDURE Customer_Top5VIP  @Year int
AS
BEGIN
DECLARE @Month int=0
WHILE @Month<12
      BEGIN
SET @Month=@Month+1;
SELECT TOP(5) Month(Invoice.CreateAt) AS Month,Customer.ID_Customer,
Customer.NameCustomer,          Customer.BirthdayCustomer,Customer.GenderCustomer,
SUM(Invoice.TotalValue) AS Revenue, COUNT(Invoice.ID_Customer) AS Times
FROM Invoice inner join Customer on Invoice.ID_Customer=Customer.ID_Customer
WHERE Month(Invoice.CreateAt)=@Month and Year(Invoice.CreateAt)=@Year
```

```
GROUP BY Month(Invoice.CreateAt), Customer.ID_Customer, Customer.NameCustomer,
Customer.BirthdayCustomer,Customer.GenderCustomer
ORDER BY COUNT(Invoice.ID_Customer) DESC
END
END

EXEC Customer_Top5VIP @Year=2019
```

Result:

	Month	ID_Customer	NameCustomer	BirthdayCustomer	GenderCustomer	Revenue	Times
1	1	KH04	Tran Minh Long	1965-09-03	male	881000	4
2	1	KH02	Tran Ngoc Han	1974-03-04	female	950600	4
3	1	KH07	Nguyen Van Tam	1971-06-04	male	344000	2
4	1	KH05	Le Nhat Minh	1950-10-03	male	243500	2
5	1	KH01	Nguyen Van A	1960-10-10	male	371200	2

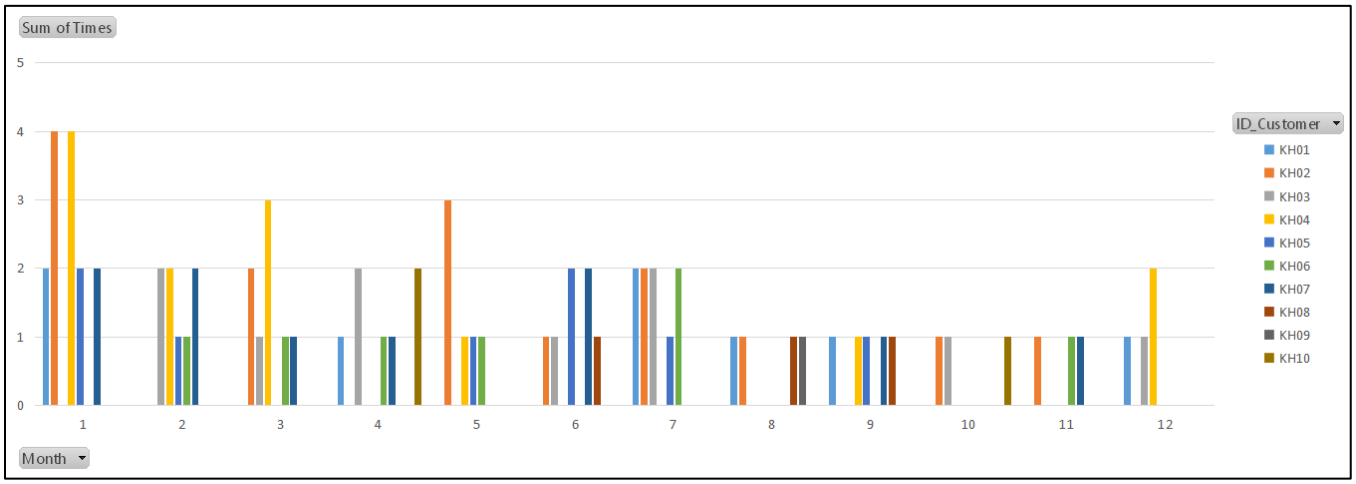
	Month	ID_Customer	NameCustomer	BirthdayCustomer	GenderCustomer	Revenue	Times
1	2	KH04	Tran Minh Long	1965-09-03	male	163500	2
2	2	KH03	Tran Ngoc Linh	1980-12-06	female	270000	2
3	2	KH07	Nguyen Van Tam	1971-06-04	male	293200	2
4	2	KH06	Le Hoai Thuong	1981-12-31	female	100000	1
5	2	KH05	Le Nhat Minh	1950-10-03	male	250000	1

	Month	ID_Customer	NameCustomer	BirthdayCustomer	GenderCustomer	Revenue	Times
1	3	KH04	Tran Minh Long	1965-09-03	male	232400	3
2	3	KH02	Tran Ngoc Han	1974-03-04	female	333400	2
3	3	KH07	Nguyen Van ...	1971-06-04	male	100000	1
4	3	KH06	Le Hoai Thuong	1981-12-31	female	180000	1
5	3	KH03	Tran Ngoc Linh	1980-12-06	female	57800	1

	Month	ID_Customer	NameCustomer	BirthdayCustomer	GenderCustomer	Revenue	Times
1	4	KH03	Tran Ngoc Linh	1980-12-06	female	174000	2
2	4	KH10	Ha Duy Lap	1983-02-05	male	429600	2
3	4	KH01	Nguyen Van A	1960-10-10	male	154200	1
4	4	KH07	Nguyen Van ...	1971-06-04	male	47200	1
5	4	KH06	Le Hoai Thuong	1981-12-31	female	473600	1

Query executed successfully. | ANBINH0777\SQLEXPRESS (12.0... | ANBINH0777\MYCOM (61) | ANOCompany | 00:00:01 | 5 rows

Report:

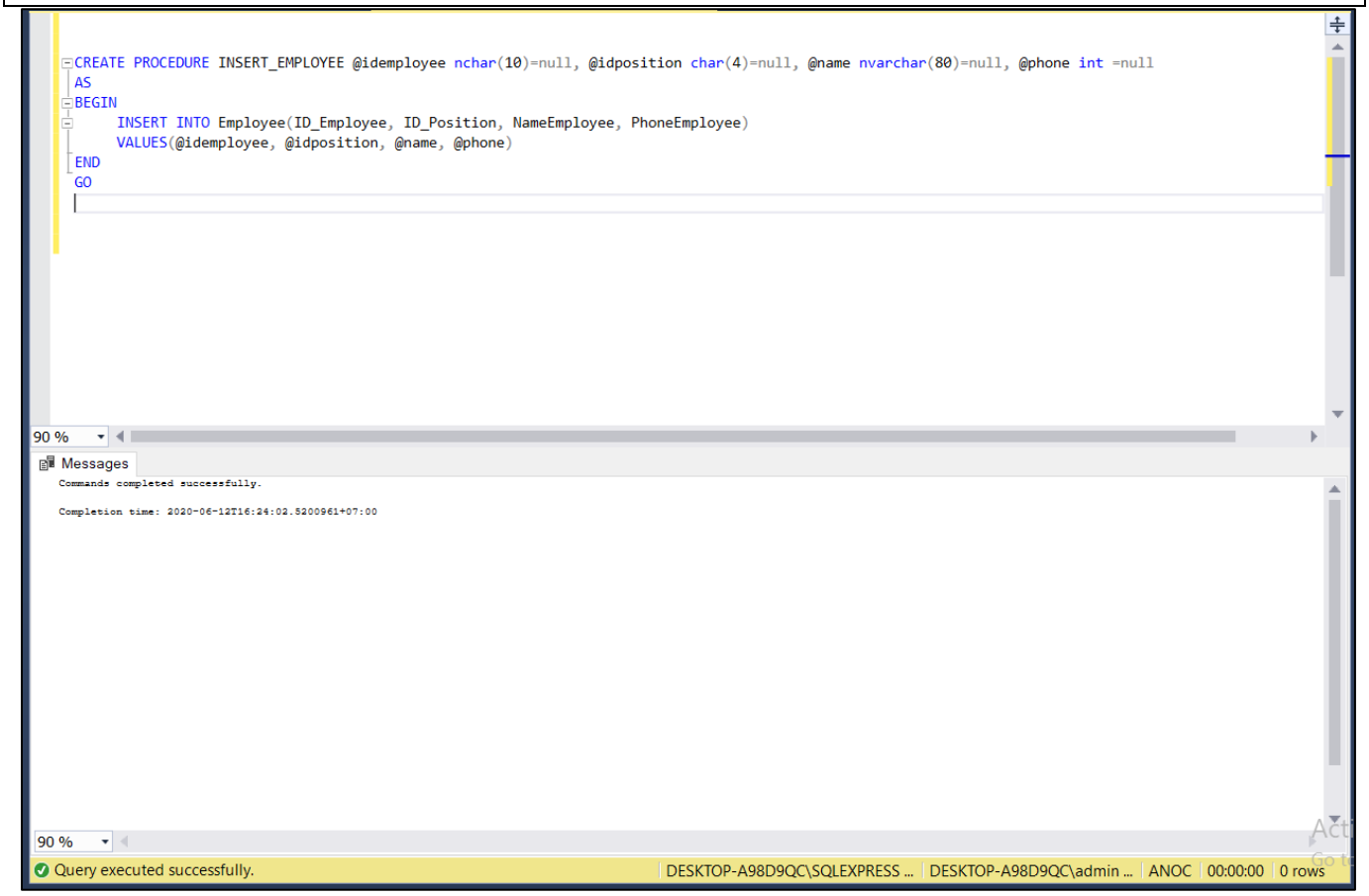


The chart shows the number of purchases made by the top 5 customers each month. The company will easily monitor customer activities and have customer appreciation policies and business strategies accordingly.

5. System management operations: Perform queries insert, delete, edit data

- Insert: The managers can use this procedure to insert 1 or many new employees.

```
CREATE PROCEDURE INSERT_EMPLOYEE @idemployee nchar(10)=null, @idposition char(4)=null, @name nvarchar(80)=null, @phone int =null
AS
BEGIN
    INSERT INTO Employee(ID_Employee, ID_Position, NameEmployee, PhoneEmployee)
    VALUES(@idemployee, @idposition, @name, @phone)
END
GO
```

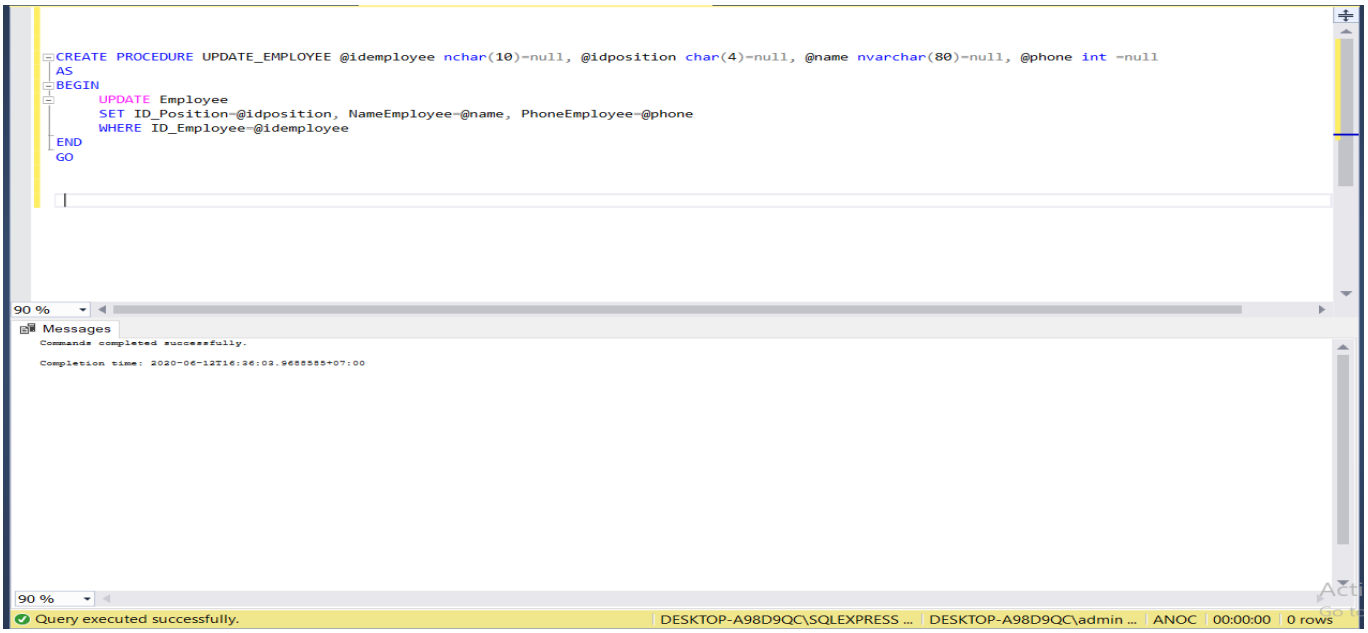


EX: Two new employess (Nguyen Thi Bo and Hiro Mot Lan) are inserted.

	ID_Employee	ID_Position	NameEmployee	PhoneEmployee
1	NVBH01	NVBH	Nguyen Nhu Nhut	927345678
2	NVBH02	NVBH	Le Thi Phi Yen	987567390
3	NVBH03	NVBH	Nguyen Van B	997047382
4	NVBH04	NVBH	Ngo Thanh Tuan	913758491
5	NVBH05	NVBH	Nguyen Thi Truc Thanh	918590387
6	NVBH100	NVBH	Nguyen Thi Bo	188345679
7	NVBH15	NVBH	Hiro Mot Lan	934068999
8	NVQL01	NVQL	Ngo Tuan	913758894
9	NVQL02	NVQL	Nguyen Thi Thanh	918590389
10	NVTK01	NVTK	Le Thanh Tuan	92275847
11	NVTK02	NVTK	Nguyen Thi Truc	918590320

- Update: The managers can use this procedure to update any attributes of an employee to manage them easily (for example when an employee change their phone number or their position)

```
CREATE PROCEDURE UPDATE_EMPLOYEE @idemployee nchar(10)=null, @idposition char(4)=null, @name nvarchar(80)=null, @phone int =null
AS
BEGIN
    UPDATE Employee
        SET ID_Position=@idposition, NameEmployee=@name, PhoneEmployee=@phone
    WHERE ID_Employee=@idemployee
END
GO
```



EX: Hiro Mot Lan ‘s PhoneEmployee has been changed
Before:

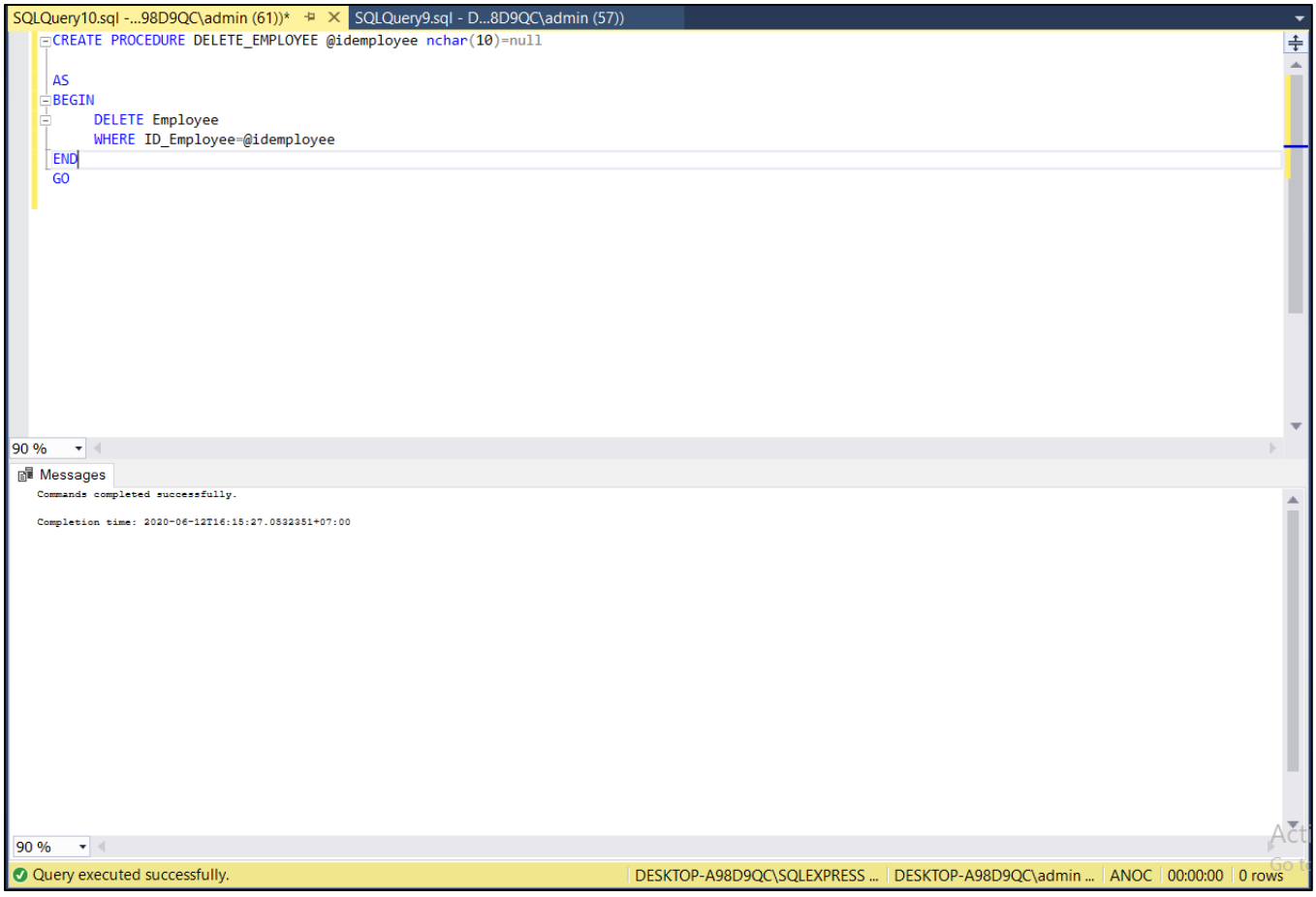
	ID_Employee	ID_Position	NameEmployee	PhoneEmployee
1	NVBH01	NVBH	Nguyen Nhu Nhut	927345678
2	NVBH02	NVBH	Le Thi Phi Yen	987567390
3	NVBH03	NVBH	Nguyen Van B	997047382
4	NVBH04	NVBH	Ngo Thanh Tuan	913758491
5	NVBH05	NVBH	Nguyen Thi Truc Thanh	918590387
6	NVBH100	NVBH	Nguyen Thi Bo	188345679
7	NVBH15	NVBH	Hiro Mot Lan	934068999
8	NVQL01	NVQL	Ngo Tuan	913758894
9	NVQL02	NVQL	Nguyen Thi Thanh	918590389
10	NVTK01	NVTK	Le Thanh Tuan	92275847
11	NVTK02	NVTK	Nguyen Thi Truc	918590320

After:

	ID_Employee	ID_Position	NameEmployee	PhoneEmployee
1	NVBH01	NVBH	Nguyen Nhu Nhut	927345678
2	NVBH02	NVBH	Le Thi Phi Yen	987567390
3	NVBH03	NVBH	Nguyen Van B	997047382
4	NVBH04	NVBH	Ngo Thanh Tuan	913758491
5	NVBH05	NVBH	Nguyen Thi Truc Thanh	918590387
6	NVBH100	NVBH	Nguyen Thi Bo	188345679
7	NVBH15	NVBH	Hiro Mot Lan	789123456
8	NVQL01	NVQL	Ngo Tuan	913758894
9	NVQL02	NVQL	Nguyen Thi Thanh	918590389
10	NVTK01	NVTK	Le Thanh Tuan	92275847
11	NVTK02	NVTK	Nguyen Thi Truc	918590320

- Delete: The managers can use the procedure to delete any employees quitting their job.

```
CREATE PROCEDURE DELETE_EMPLOYEE @idemployee nchar(10)=null

AS
BEGIN
    DELETE Employee
  WHERE ID_Employee=@idemployee
END
GO
```

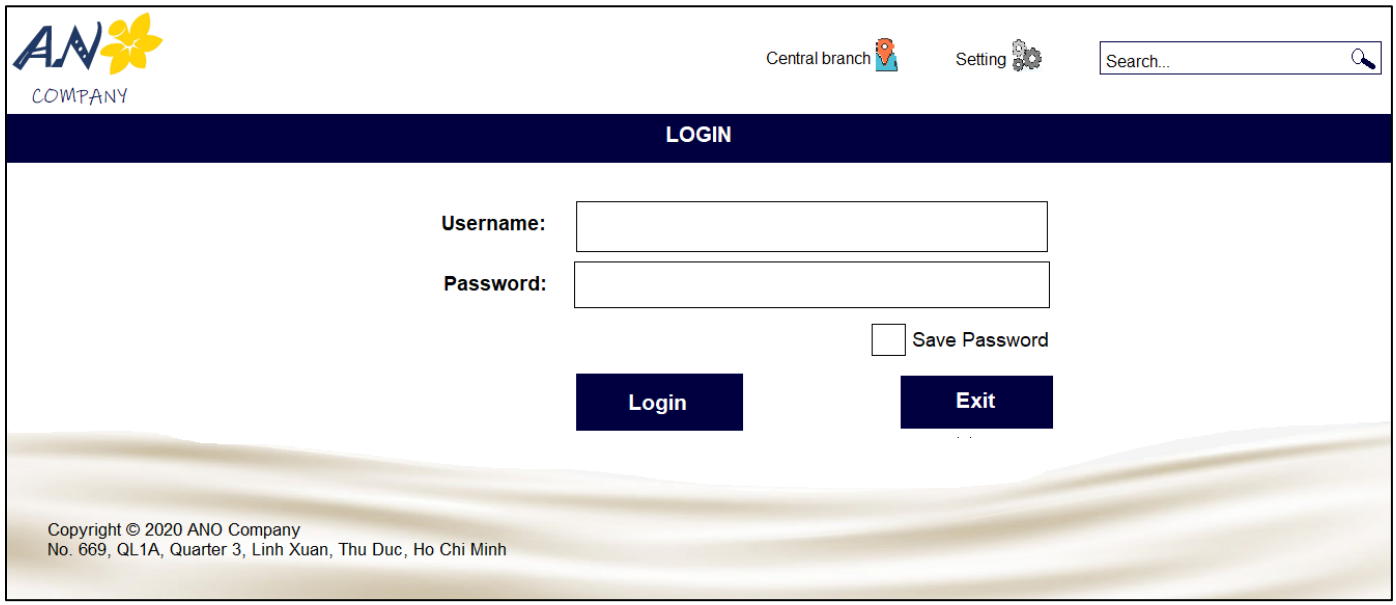


CHAPTER 4: BUILDING MANAGEMENT APPLICATION

1.Functional diagram

2.Building application program

2.1. Main screen

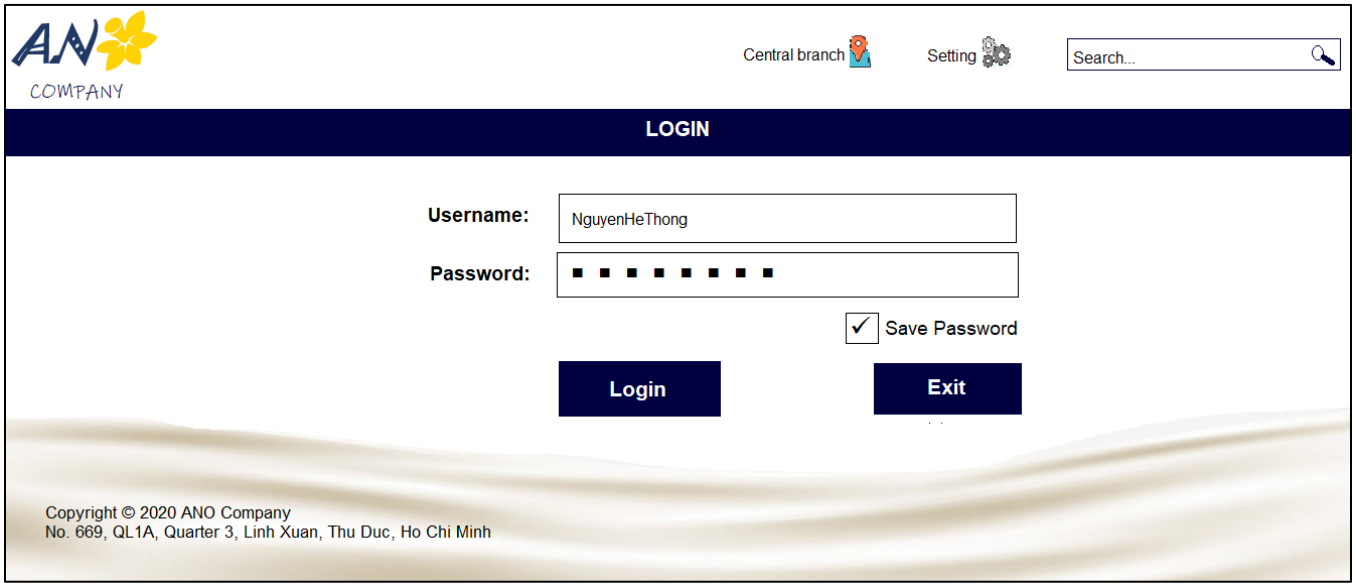


No.	Name	Type	Meaning
1	Central branch	Button	Navigate to a Google Map website to direct users to the right company address.
2	Setting	Button	Allow users to change system color.
3	Search	Textbox	Allow users to search related information
4	Username	Textbox	Allow users to enter Username by yourself
5	Password	Textbox	Allow users to enter Passwword by yourself
6	Save Password	Checkbox	Allow users to save Username and Passwword by yourself in sales management software
7	Login	Button	Clicks this button to execute the permission
8	Exit	Button	Exit ANO Company sales management software

First, the user will enter the box Username: and Password: . Please make sure you have entered the correct Username, Password and click Login. The system will start checking with the data about the user logged in the system. If they match, you will be moved to a new interface and allowed to perform operations related to the functions of each employee and each user.

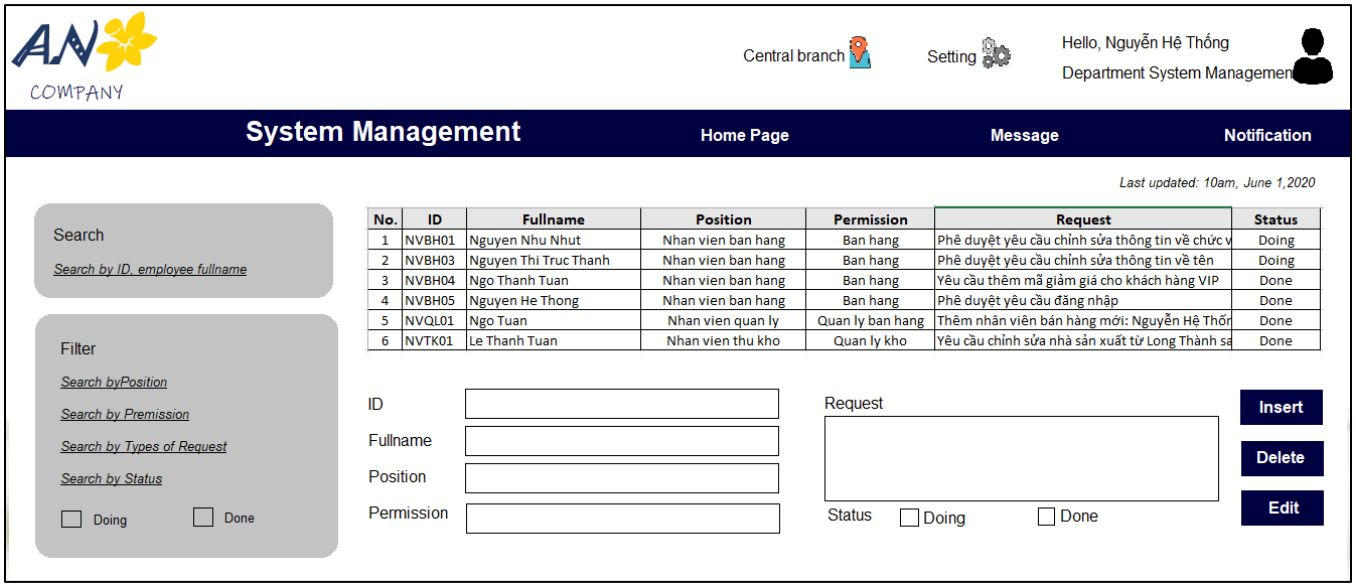
There is a function Save Password to help employees save passwords, which helps to avoid forgetting passwords and reset passwords for employees.

If you want to temporarily stop working, then click **Exit** on the main screen to exit. After, screen when you have entered Username, Password and tick Save Password.



2.2 Working screen of the Department System

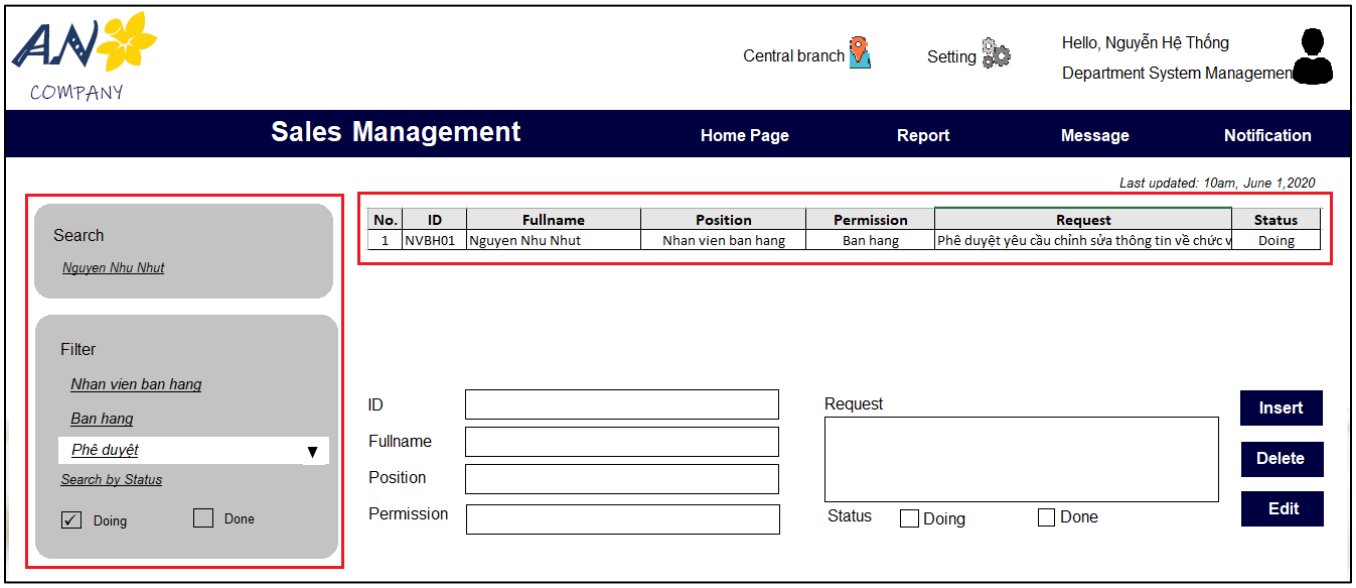
If the user is an employee of the System Office, after click **Login** this is their main working interface.



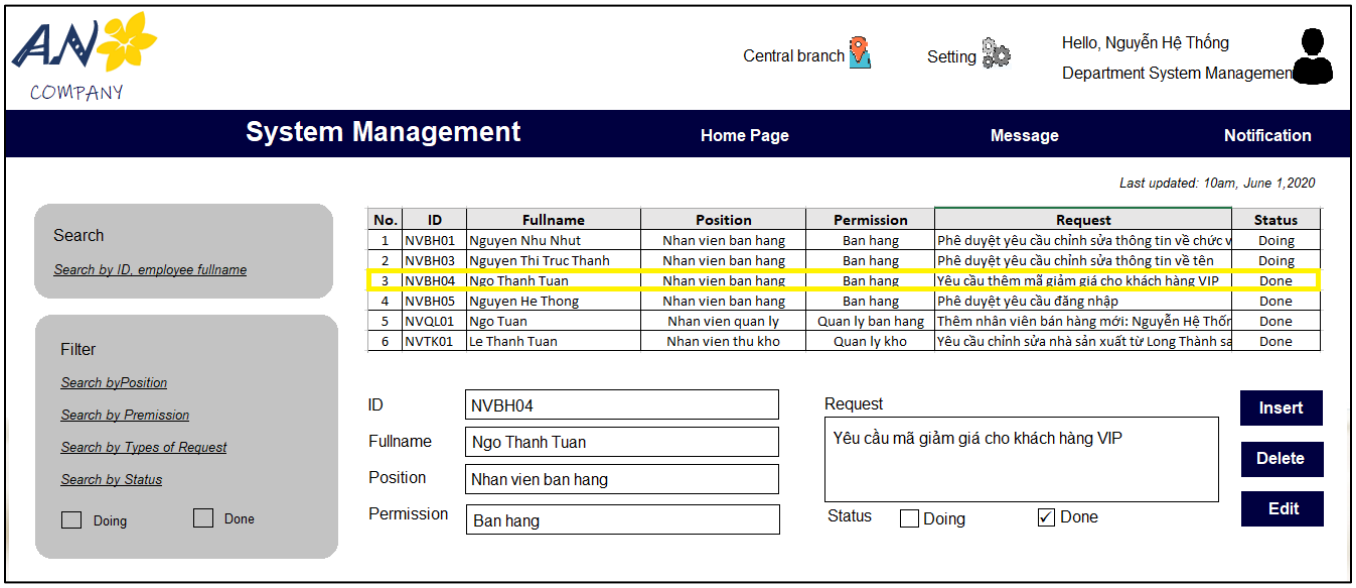
No.	Name	Type	Meaning
1	Fullname Employee	Label	Determine the employee's name again
2	Department	Label	Determine the employee's department again
3	Home Page	Button	Help employee can return Main Screen

4	Message	Button	Enables the exchange of information between the relevant departments
5	Notification	Button	Internal communication is allowed
6	System Management	DataGird	Displays all information about the operation on the management system to the System Room staff
7	Search by (ID, Fullname Employee, Position, Premission, Type)	Textbox	Help employee search information fast and accurate
8	Search by Type	Combobox	Lets choose Types
9	Search by Satus	Checkbox	Lets choose between Doing and Done
10	ID, Fullname Employee, Position, Premission, Type	Textbox	Allows users to view and execute the Insert, Delete, Edit commands from the keyboard
11	Satus	Checkbox	Display and choose between Doing and Done
12	Insert, Delete, Edit	Button	Perform Insert, Delete, Edit of the activity board on the system from the keyboard

With the screen above, when performing the Search and Filter by entering the relevant information, the DataGird table will automatically be updated and display the information that employees need. Below, is the screen when executing Search and Filter commands.

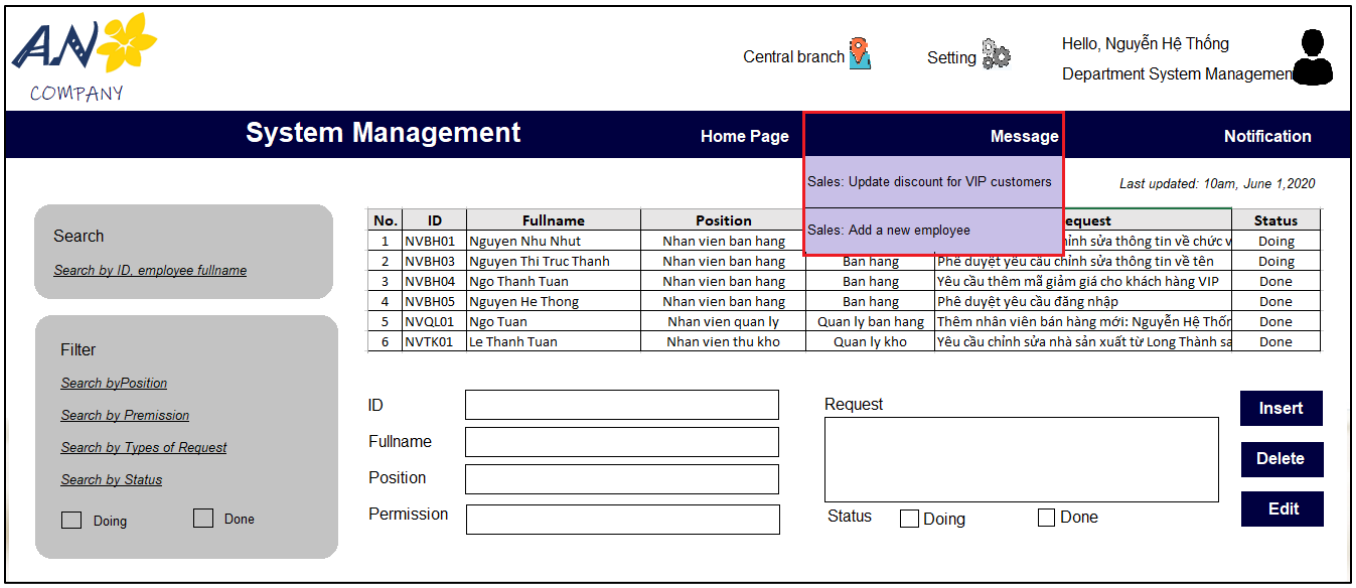


When selecting a row in the system activity panel, all information will be displayed in ID, Fullname Employee, Position, Premission, Request, Status below, as shown below.

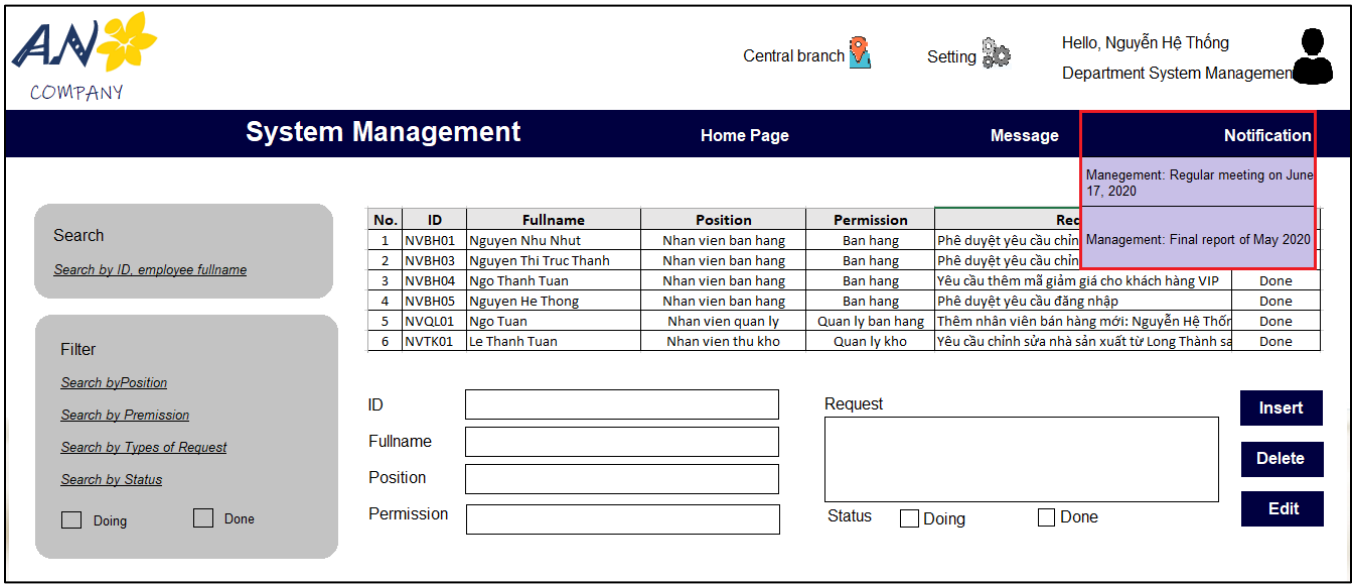


Here, employees can perform Insert, Delete, Edit the request on the buttons: **Insert** , **Delete** , **Edit** .

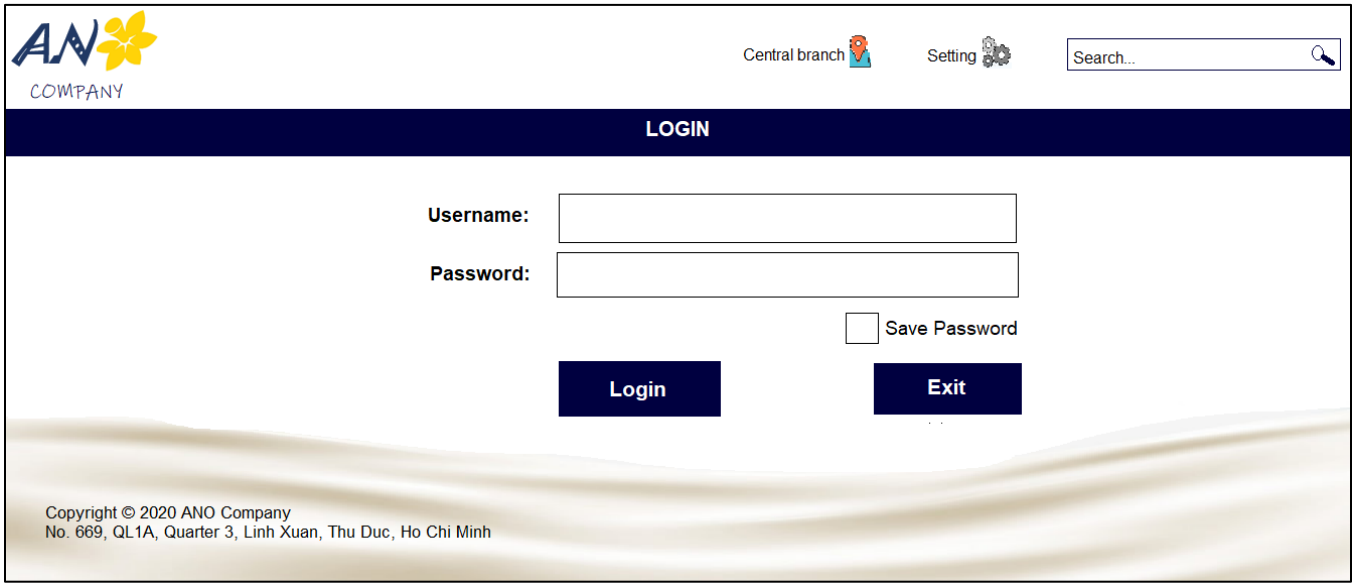
When clicking on **Message** it will display the message as shown below.



When clicking on **Notification** it will display the message as shown below.



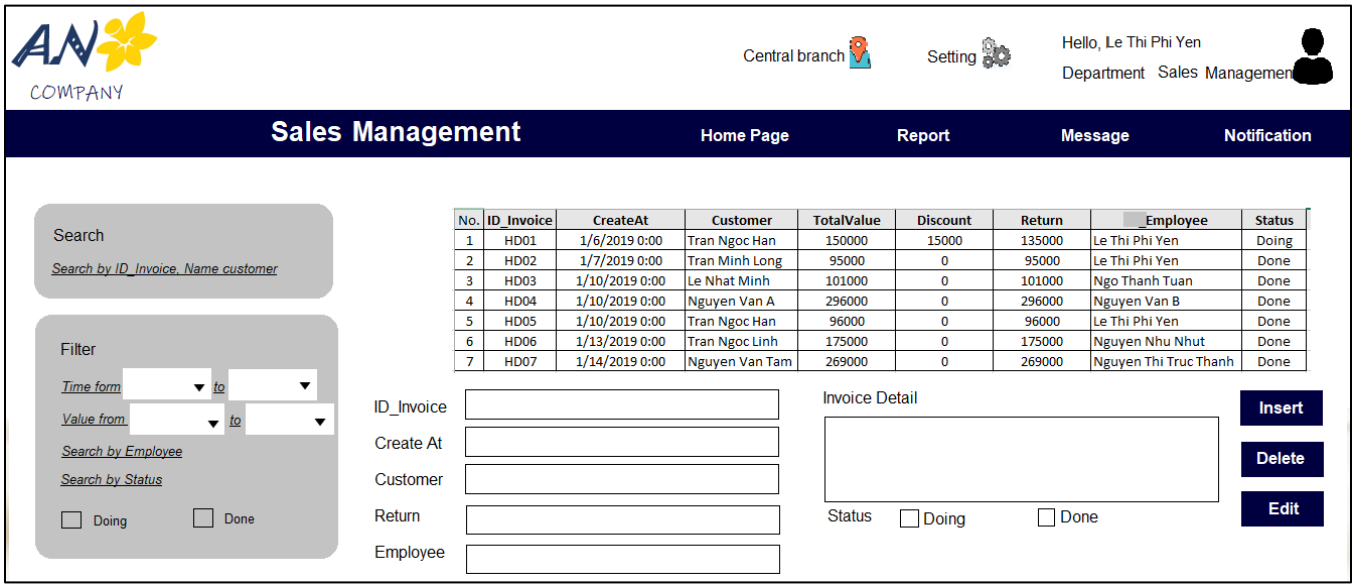
If you want to exit the program, you must select **Home Page**. This is the screen when done **Home Page**



Execute the command **Exit** as mentioned above to exit ANO Company sales management software.

2.3 Working screen of the Department Sales

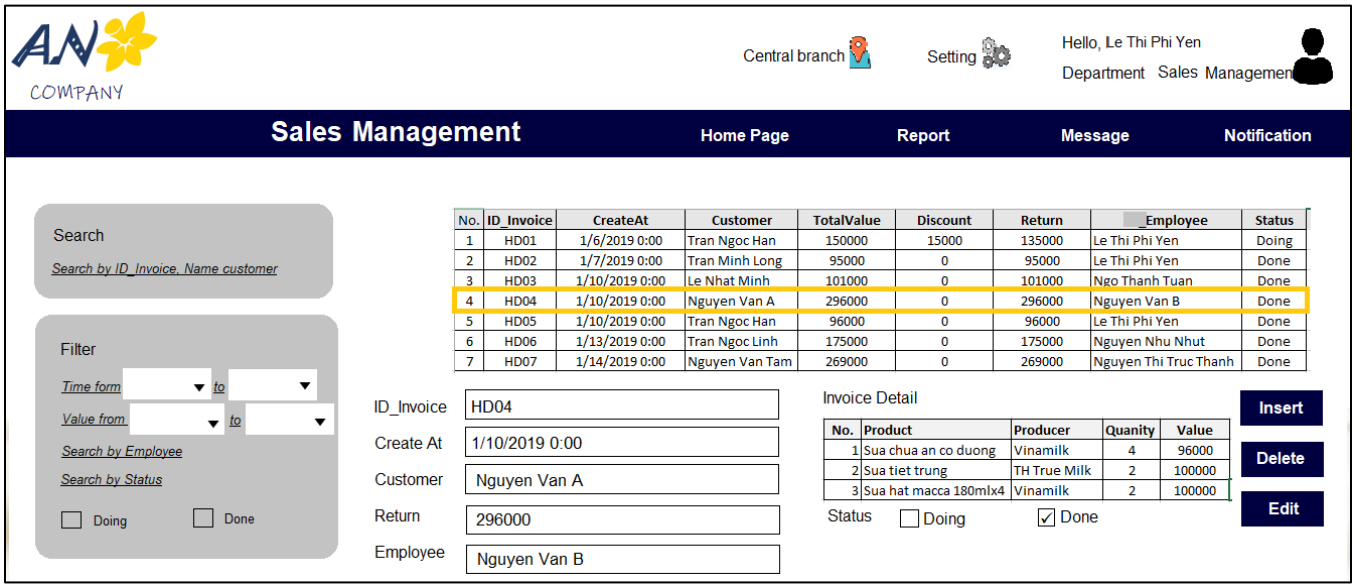
If the user is an employee of the Sales Office, after click **Login** this is their main working interface.



No.	Name	Type	Meaning
1	Fullname Employee	Label	Determine the employee's name again
2	Department	Label	Determine the employee's department again
3	Home Page	Button	Help employee can return Main Screen
4	Report	Button	Make reports quickly
5	Message	Button	Enables the exchange of information between the relevant departments
6	Notification	Button	Internal communication is allowed
7	Sales Management	DataGird	Displays all information about sale process to the Sales staff
8	Search by (ID, Employee, Status)	Textbox	Help employee search information fast and accurate
9	Search by Time	Textbox	Search information with a determined time
10	Search by Value	Textbox	Search information with a determined value
11	Search by Satus	Checkbox	Lets choose between Doing and Done
12	ID, Create At, Customer, Employee, Return	Textbox	Allows users to view and execute the Insert, Delete, Edit commands from the keyboard
13	Invoice Detail	DataGird	Clearly display the items and quantities sold in that invoice
14	Satus	Checkbox	Display and choose between Doing and Done
15	Insert, Delete, Edit	Button	Perform Insert, Delete, Edit of the activity board on the system from the keyboard

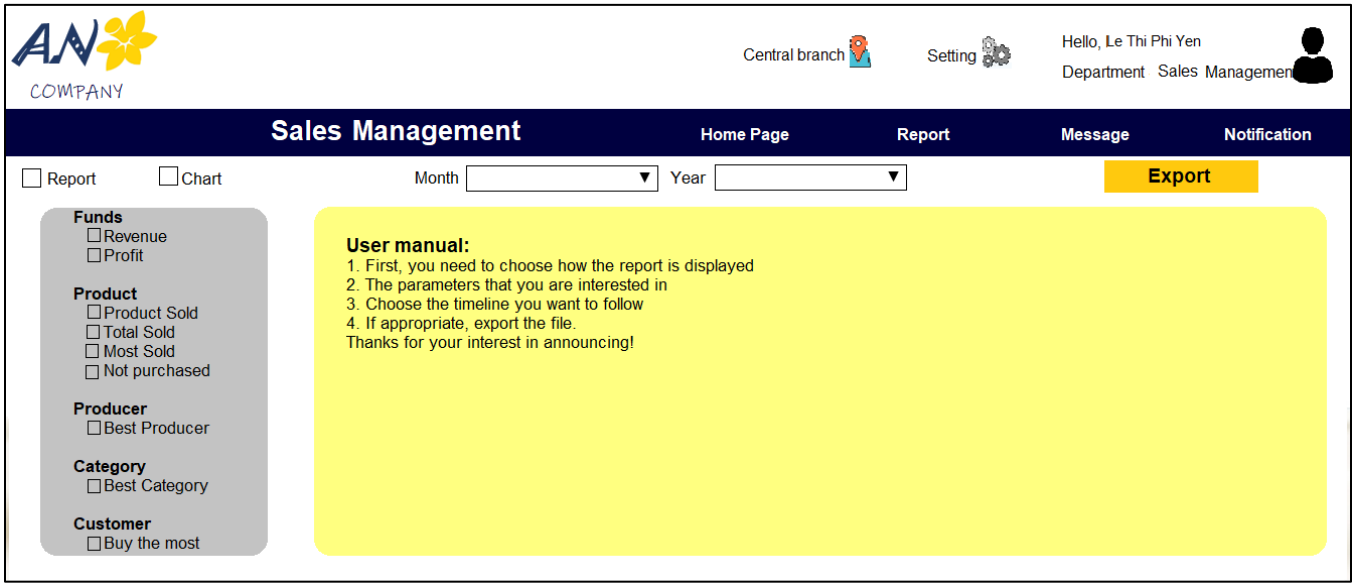
The Home Page, Message, Notification, Search, Filter is the same as the previous one mentioned.

When selecting a row in the system activity panel, all information will be displayed in ID, Create At, Customer, Employee, Return, Status below, as shown below.



Here, employees can perform Insert, Delete, Edit the request on the buttons: **Insert** , **Delete** , **Edit** .

In this interface, there is also a new Button **Report**. This will help employees easily summarize reports and observe the business situation in the most detailed way. The screen works as follows.



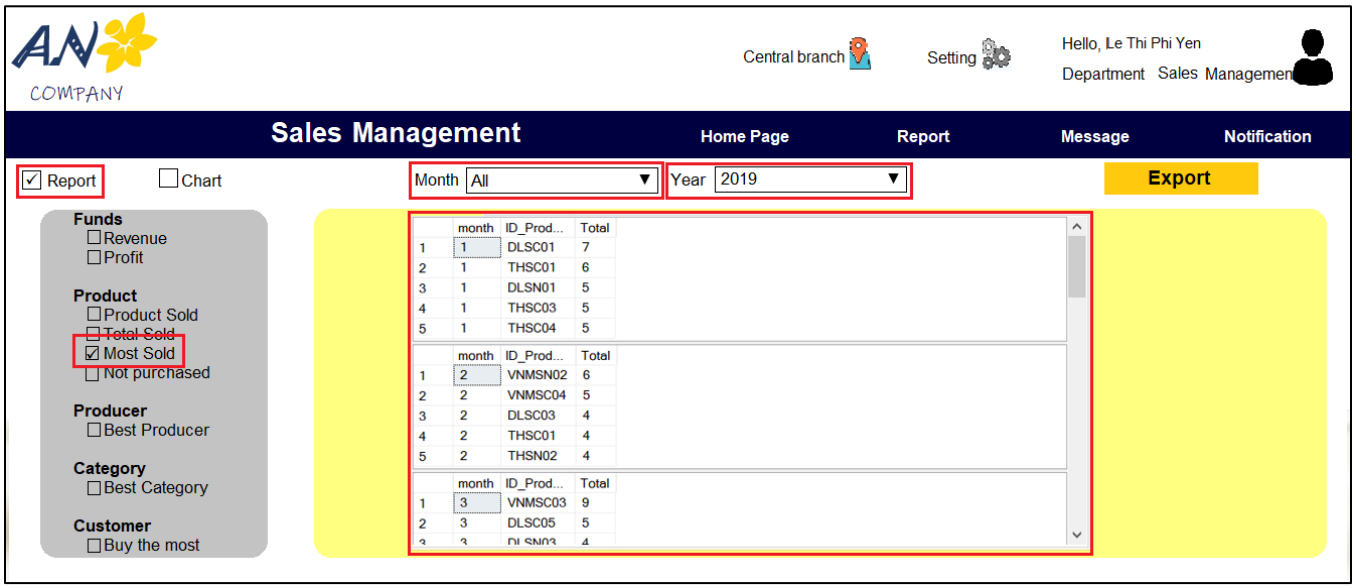
No.	Name	Type	Meaning
1	Display mode	Checkbox	Allow employees to choose Report or Chart to Export file
2	Revenue, Profit, Product Sold, Total Sold, Most Sold, No Purchased, Best Producer, Best Category, Buy The Most	ComboBox	Allow employees to select relevant parameters for reporting
3	Time (Month, Year)	Textbox	Search information with a determined time
4	Export	Button	Allow employees to export data file

Please read the following instructions carefully and follow them correctly.

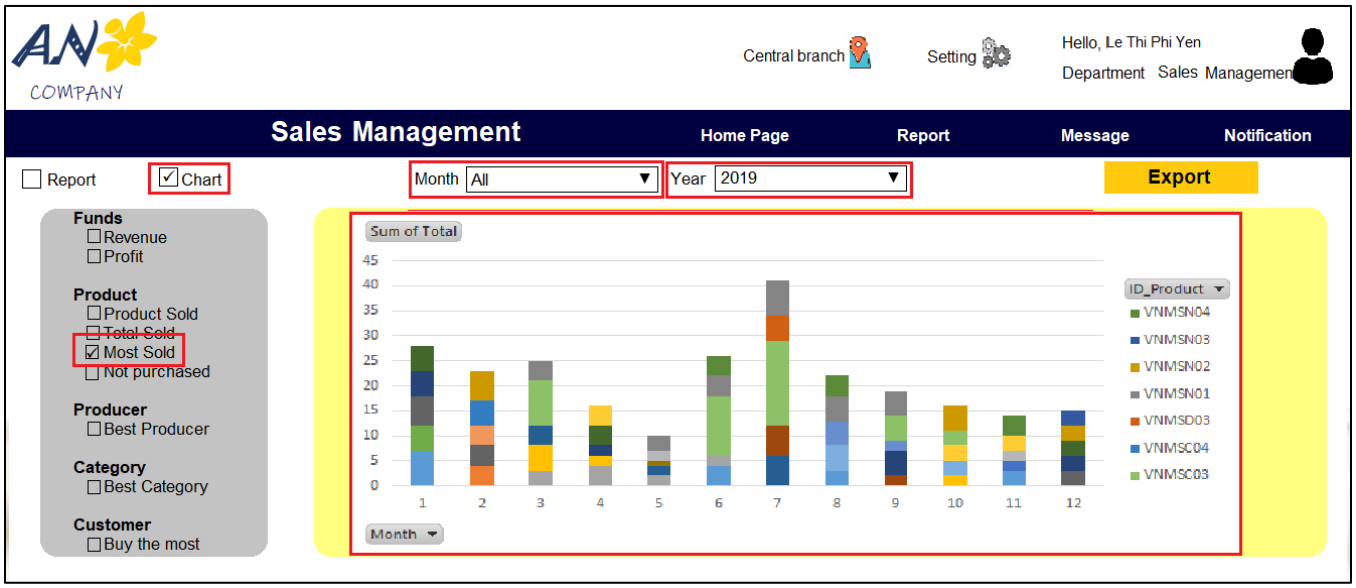
User manual:
1. First, you need to choose how the report is displayed
2. The parameters that you are interested in
3. Choose the timeline you want to follow
4. If appropriate, export the file.

Thanks for your interest in announcing!

For example, Empoyee want to export file about Top 5 the most sold products in the month and in the year 2019 to help managers manage business strategies.

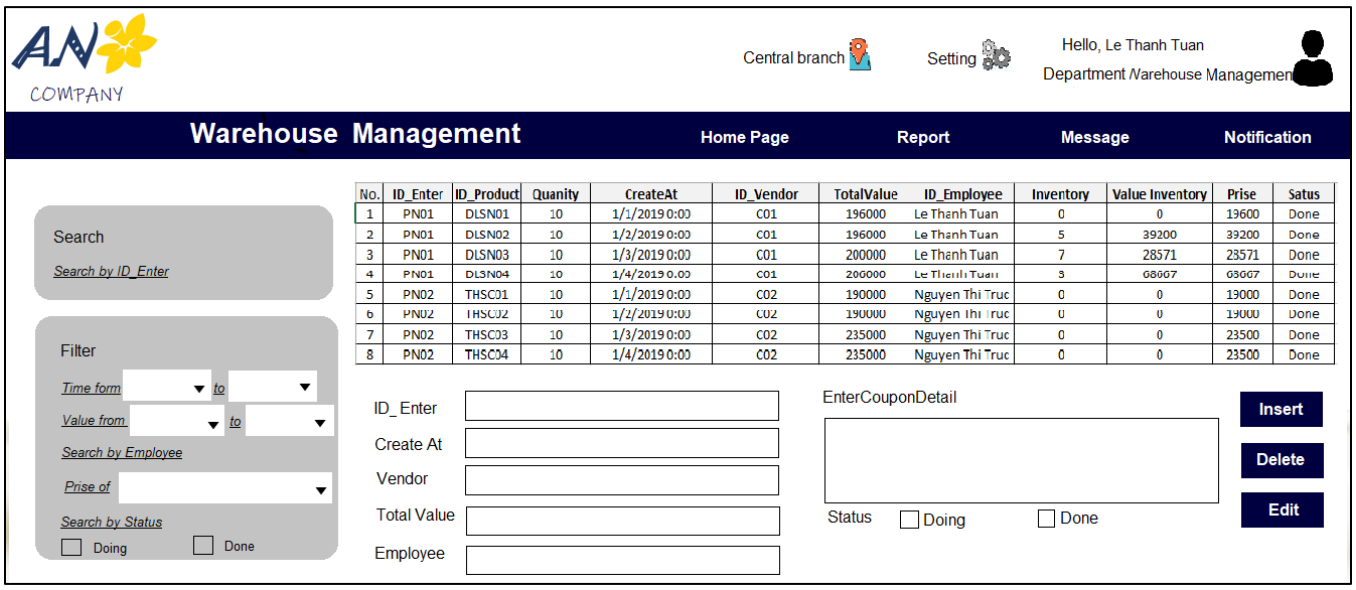


If you select the chart display mode, it will be as follows.



When the mining needs are met, click  to export as Excel  to your computer.

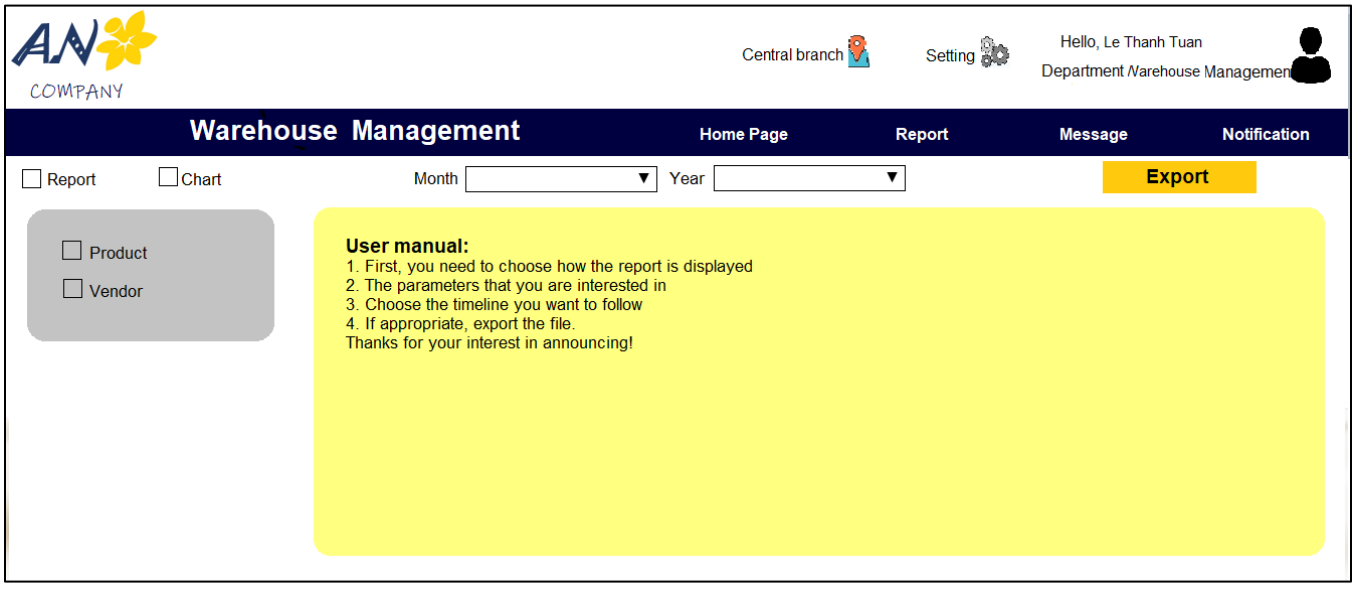
2.3 Working screen of the Department Warehouse



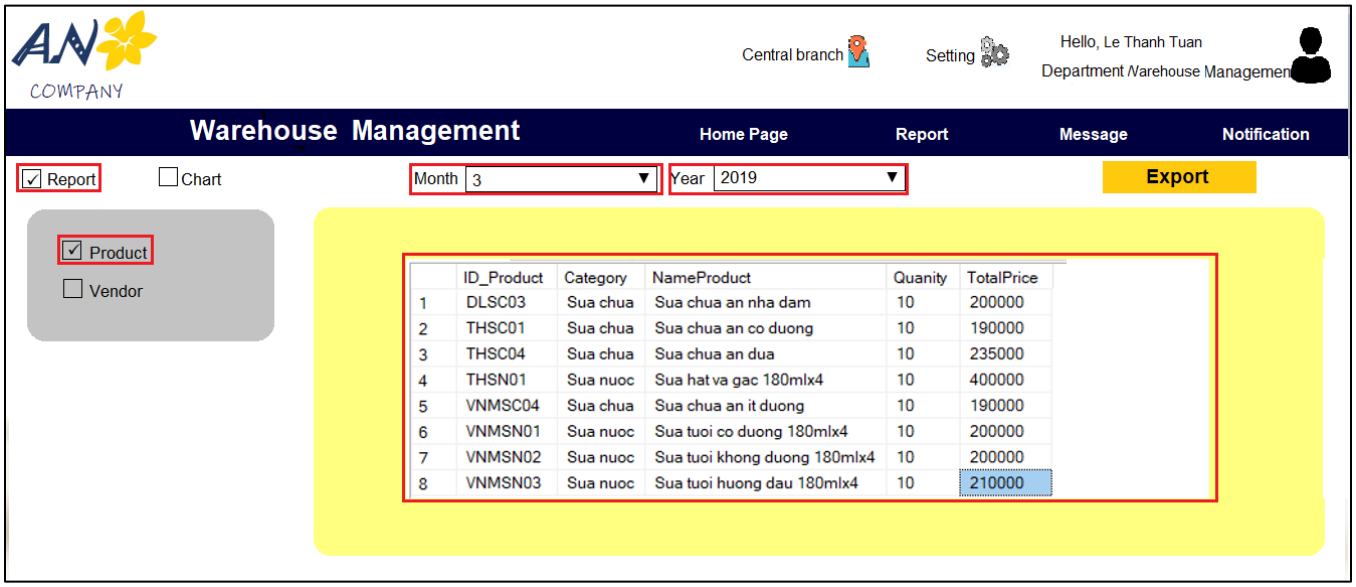
The Home Page, Message, Notification, Search, Filter is the same as the previous one mentioned.

When selecting a row in the system activity panel, all information will be displayed in ID, Create At, Vendor, Total Value, Employee, Enter Coupon Detail, Status below. Here, employees can perform Insert, Delete, Edit the request on the buttons: Insert, Delete, Edit.

In this interface, if click Button Report. This will help employees easily summarize reports and observe the business situation in the most detailed way. The screen works as follows.

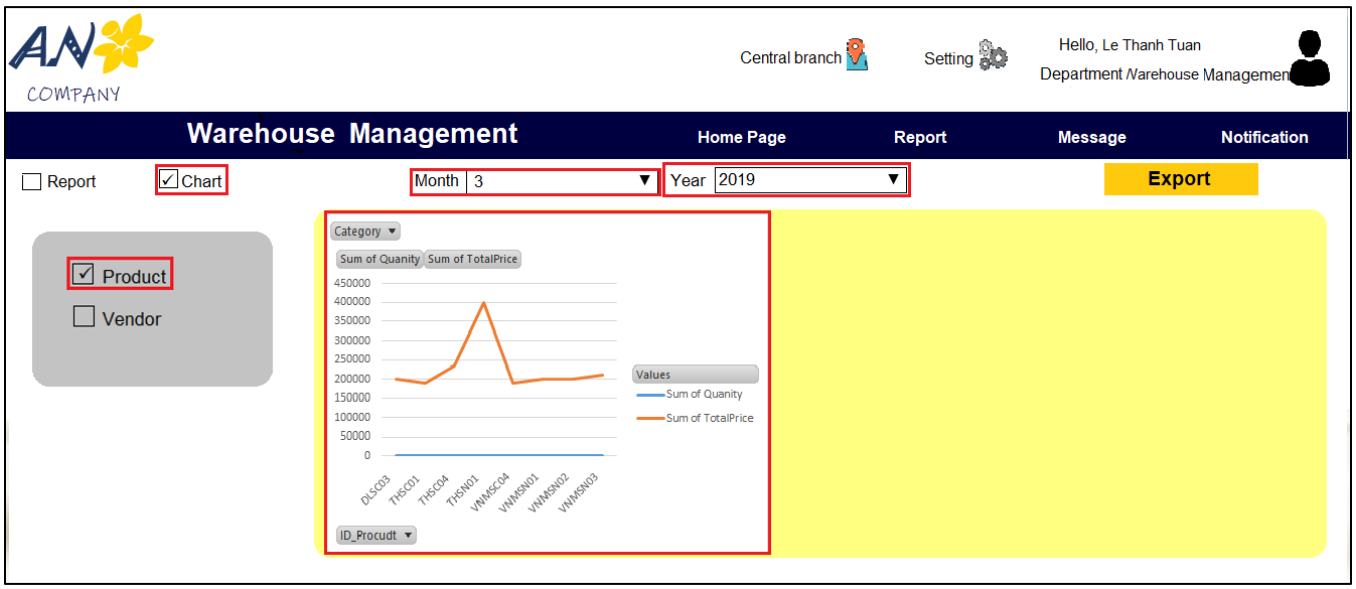


For example, employee want to Create a list of imported items in month 3, 2019 to help the warehouse department check stock status.



	ID_Product	Category	NameProduct	Quanity	TotalPrice
1	DLSC03	Sua chua	Sua chua an nha dam	10	200000
2	THSC01	Sua chua	Sua chua an co duong	10	190000
3	THSC04	Sua chua	Sua chua an dua	10	235000
4	THSN01	Sua nuoc	Sua hat va gac 180mlx4	10	400000
5	VNMSC04	Sua chua	Sua chua an it duong	10	190000
6	VNMSN01	Sua nuoc	Sua tuoi co duong 180mlx4	10	200000
7	VNMSN02	Sua nuoc	Sua tuoi khong duong 180mlx4	10	200000
8	VNMSN03	Sua nuoc	Sua tuoi huong dau 180mlx4	10	210000

If you select the chart display mode, it will be as follows.

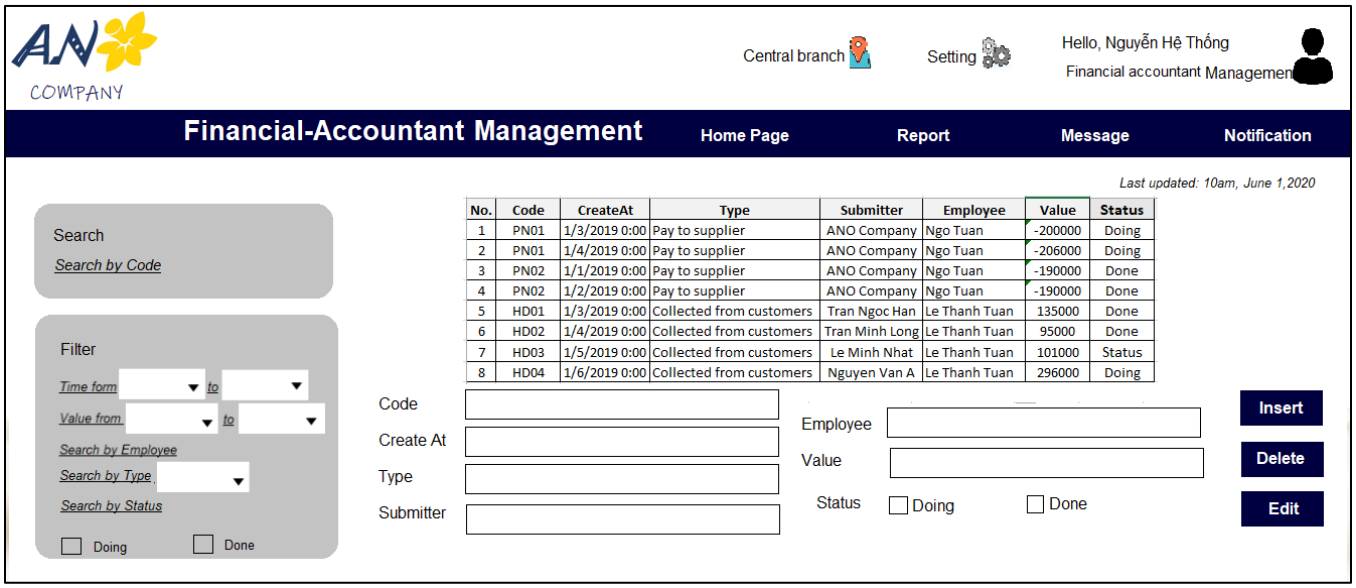


When the mining needs are met, click Export to export as Excel 

to your computer.

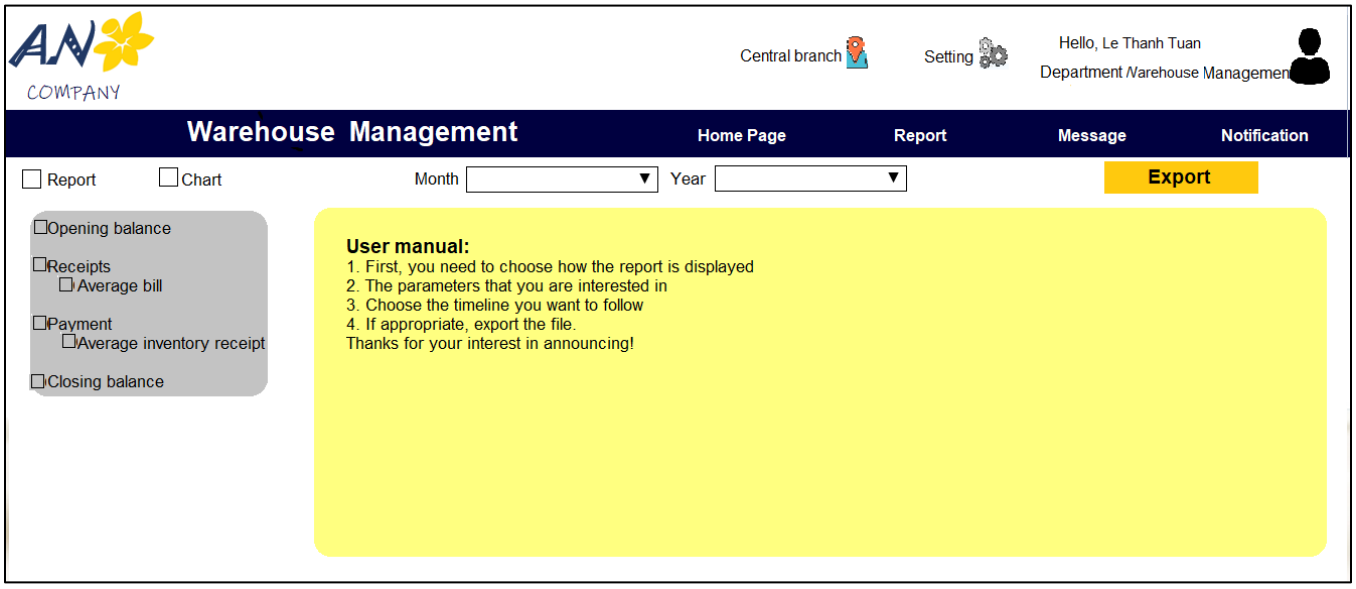
2.4 Working screen of the Department Financial-Accounting

The **Home Page**, **Message**, **Notification**, Search, Filter is the same as the previous one mentioned.

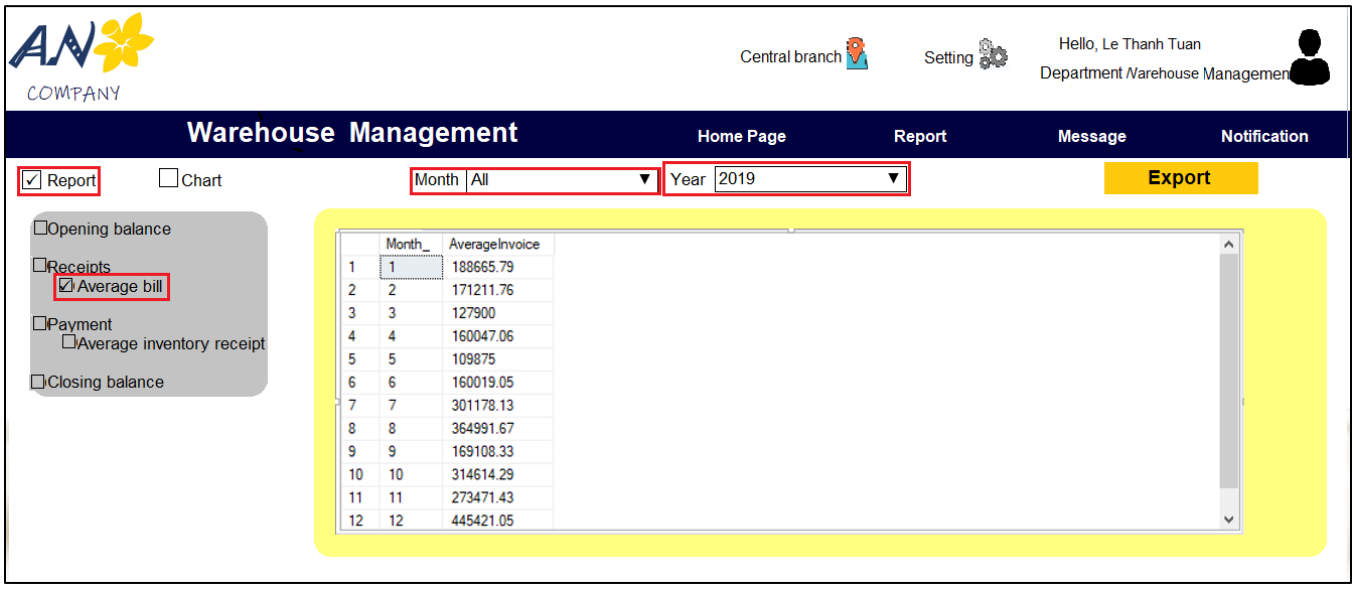


When selecting a row in the system activity panel, all information will be displayed in Code, Create At, Type, Submitter, Value, Employee, Status below. Here, employees can perform Insert, Delete, Edit the request on the buttons: **Insert**, **Delete**, **Edit**.

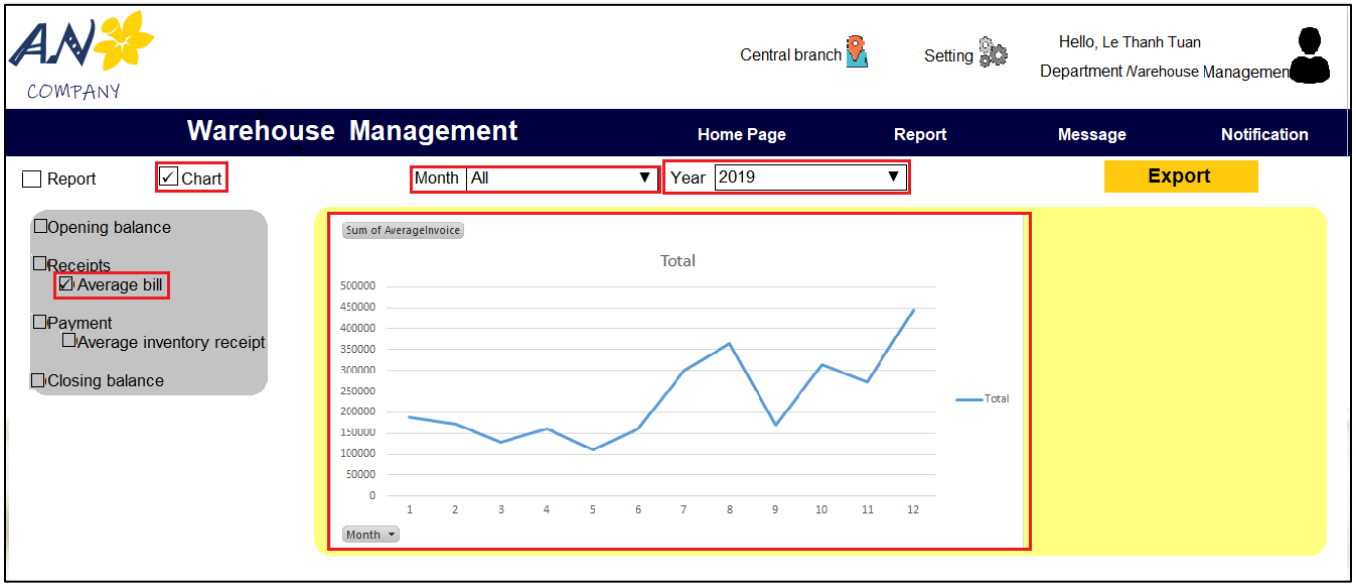
In this interface, if click Button **Report**. This will help employees easily summarize reports and observe the business situation in the most detailed way. The screen works as follows.



For example, An enterprise only exists when its goods and products are exchanged and traded in the market, ANO Company is also among them. For the exchange to be effective, the company needs to carefully monitor its sales. In addition, the company should combine with market research and conduct reasonable price adjustment strategies. The average value of all invoices sold in each month of the year is a prerequisite for implementing this strategy.



If you select the chart display mode, it will be as follows.



When the mining needs are met, click Export to export as Excel 

to your computer.

CHAPTER 5: CONCLUDE

1.Results of the topic

Through the implementation of the project, the team was aware of the steps to take when building a complete database from going from the lowest design and construction steps such as defining requirements, ERD, Logical design. Diagram, Physical Diagram until data binding.

With the newly built database, the group has proficiently manipulated SQL statements to query data and tabulate data. To make the tables meaningful, the team learned how to use a PivotTable to be able to visualize the datasheet, giving an overview to administrators. From there, the management will have reasonable and specific strategies to run the company well and improve its market share in the milk distribution market.

Due to the situation of the Covid-19 epidemic, the team developed and made a report on both channels: Online and Offline to ensure the report was completed on schedule. Thereby, teamwork skills are enhanced both indirect work and online work.

2.Limitations of the topic

Because English is limited, the presentation of the contents of the project takes a lot of time. Finding out more about foreign documents is difficult.

Initially build a database prone to mistakes burning phase. The team burned the first phase of database design, so when it came to executing the query, the team had a hard time. That forced the group to rebuild a new database and it took quite a long time.

The subject is not creative, groundbreaking. The topic revolves around Sales operations, Warehouse management operations, Financial-accounting operations, System management operations. These are the basic tasks in sales.

The time of doing the project is too long, so the members have a depressed attitude and do not urge at work. Besides, due to the effects of the COVID-19 epidemic, the group did not have many direct exchanges, so the exchange of information to build the project was not effective (network connection, working environment, ...).

2. Development direction of the topic

For the management and support of decision making to be effective, the topic is towards building an app on the phone and the computer so that employees and management can easily track and update information.

For the business process to be effective, the group will deploy and implement some new related operations:

Warehouse management operations: combined with sales data to implement an effective warehousing strategy, ensuring that goods are not overdue before leaving the warehouse. This helps protect the reputation and quality of the company's goods.

Human resource operations: to closely manage each employee, review the strengths and weaknesses of each employee, and create a specific training plan. That helps to improve the personnel level of the company.

Marketing business: combining Sales data with CRM data to make important decisions to reach more customers, expand markets, and increase market share.