**Харківський Технологічний Університет «ШАГ»**

**Кафедра Розробки програмного забезпечення**

**КУРСОВИЙ ПРОЕКТ**

**“ База даних для магазину техніки”**

**Студента групи 216**

**Тимченка О. В.**

**Науковий керівник:**

**Задерей Ю.М.**

**Харків 2022**

**Зміст**

**Вступ……………………………………………………………….……..3**

**Технічне завдання ……...………………………………………………4**

**1. Структура бази даних……………………………………...…...…..5**

**1.1Таблиці …………………………………….………………..5**

**1.2Схема даних…….…………………………………………10**

**2.Запити………………………………………………………………….14**

**2.1 Уявлення……………………………………………………..14**

**2.2 Функції………………………………………………………18**

**2.3 Збережені процедури...……………….……………..…22**

**2.4 Тригери……………………………………..………………..27**

**3.Ролі………………………………………………………………………29**

**Висновки………………………………..……………………………32**

**Список використаної літератури................................................ 33**

**Вступ**

У наші часи володіння певною інформацією відіграє велику роль,

будь-яка інформація, від прогнозу погоди до замовлення в інтернеті варто зберігати. Тому потрібні інструменти та застосунки які дозволяють ефективно зберігати та використовувати інформацію.

Саме з цієї причини бази даних є невід'ємною частиною будь чого, починаючи з маленьких магазинів і закінчуючи великими державними структурами.

**Технічне завдання**

**Тема:** База даних магазину техніки

**Технології:** MSSQL, Microsoft SQL Server Management Studio 18

**Опис функціональності:** зберігання та обробка даних магазину

**Перелік основних функцій бази даних**

1. Зберігання всіх даних за товарами.
2. Зберігання даних робітників.
3. Зберігання даних клієнтів.
4. Пошук товару за країною виробником.
5. Пошук товару за брендом.
6. Формування усіх замовлень за останній місяць.
7. Підрахунок прибутку за останній місяць.
8. Відміна товару в замовленні.
9. Відміна замовлення.
10. Підрахунок кількості актуальних пропозицій.

**Структура бази даних**

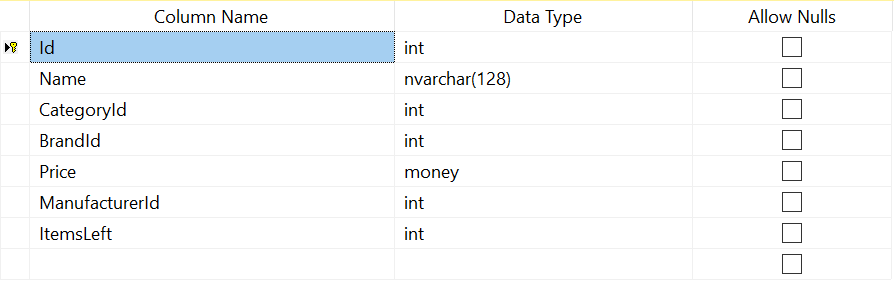
**Таблиці**

Моя база даних складається з 11 таблиць:

* Items
* Brands
* Manufacturers
* Categories
* Departments
* Employees
* Positions
* Sells
* SellsOrders
* Orders
* Clients

**Items:**

Ця таблиця містить 7 полів, як показано на зображенні 1.1.1:

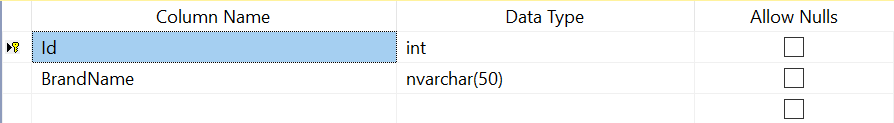
****

**Зображення 1.1.1 Поля таблиці Items**

* Поле Id є первинним ключем та лічильником товарів.
* Поле Name є назвою товара.
* Поле CategoryId є зовнішнім ключем для таблиці Categories та містить Id категорії товару.
* Поле BrandId є зовнішнім ключем для таблиці Brands та містить Id бренду товару.
* Поле Price містить ціну товару.
* Поле ManufacturerId є зовнішнім ключем для таблиці Manufacturers та містить Id країни виробника товару.
* Поле ItemsLeft містить кількість товару, що залишилося на складі.

**Brands:**

Ця таблиця містить 2 поля, як показано на зображенні 1.1.2:

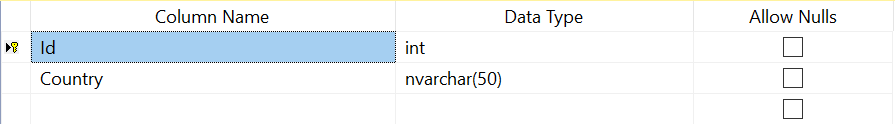
****

**Зображення 1.1.2 Поля таблиці Brands**

* Поле Id є первинним ключем та лічильником брендів товарів.
* Поле BrandName є назвою бренда.

**Manufacturers:**

Ця таблиця містить 2 поля, як показано на зображенні 1.1.3:

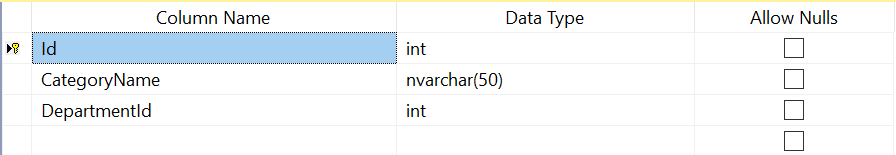
****

**Зображення 1.1.3 Поля таблиці Manufacturers**

* Поле Id є первинним ключем та лічильником країн виробників товарів.
* Поле Country містить назву країни.

**Categories:**

Ця таблиця містить 3 поля, як показано на зображенні 1.1.4:

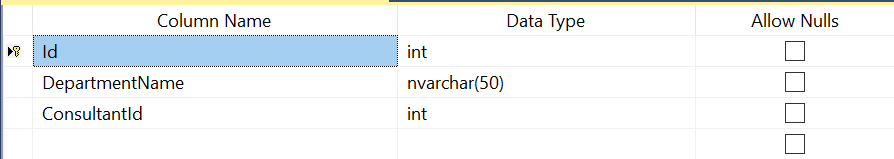
****

**Зображення 1.1.4 Поля таблиці Categories**

* Поле Id є первинним ключем та лічильником категорій товарів.
* Поле CategoryName містить назву категорії товарів.
* Поле DepartmentId є зовнішнім ключем для таблиці Departments та містить Id відділу товара.

**Departments:**

Ця таблиця містить 3 поля, як показано на зображенні 1.1.5:

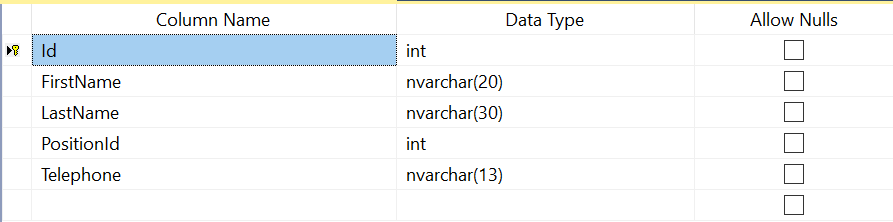
****

**Зображення 1.1.5 Поля таблиці Departments**

* Поле Id є первинним ключем та лічильником відділу товарів.
* Поле DepartmentName містить інформацію про назву відділу.
* Поле ConsultantId є зовнішнім ключем для таблиці Employees та містить Id робітника, що є консультантом по певному відділу.

**Employees:**

Ця таблиця містить 5 полів, як показано на зображенні 1.1.6:

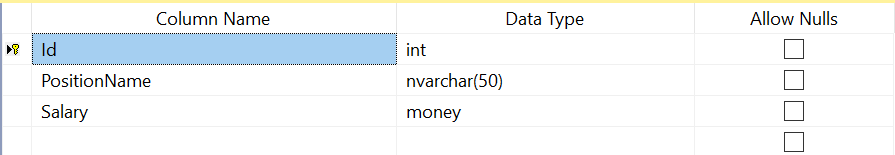
****

**Зображення 1.1.6 Поля таблиці Employees**

* Поле Id є первинним ключем та лічильником робітників магазину.
* Поле FirstName містить ім’я робітника.
* Поле LastName містить фамілію робітника.
* Поле PositionId є зовнішнім ключем для таблиці Positions та містить Id посади робітника.
* Поле Telephone містить номер робітника.

**Positions:**

Ця таблиця містить 3 поля, як показано на зображенні 1.1.7:

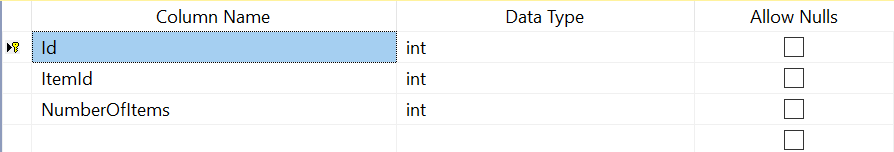
****

**Зображення 1.7 Поля таблиці Positions**

* Поле Id є первинним ключем та лічильником посад робітників магазину.
* Поле PositionName містить назву посади робітника.
* Поле Salary містить зарплату для певної посади.

**Sells:**

Ця таблиця містить 3 поля, як показано на зображенні 1.1.8:

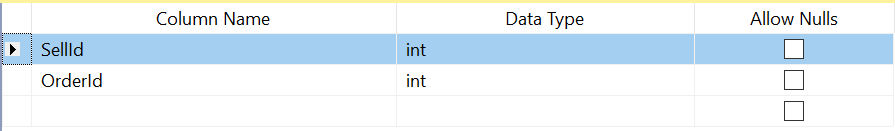
****

**Зображення 1.1.8 Поля таблиці Sells**

* Поле Id є первинним ключем та лічильником продаж певного товару.
* Поле ItemId є зовнішнім ключем для таблиці Items та містить Id певного товару.
* Поле NumberOfItems містить інформацію про кількість одиниць товару, що продається.

**SellsOrders:**

Ця таблиця містить 2 поля, як показано на зображенні 1.1.9:

****

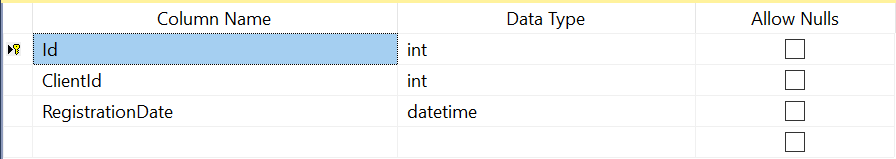
**Зображення 1.1.9 Поля таблиці SellsOrders**

* Поле SellId є зовнішнім ключем до таблиці Sells.
* Поле OrderId є зовнішнім ключем до таблиці Orders.

Разом ці поля формують складений первинний ключ.

**Orders:**

Ця таблиця містить 3 поля, як показано на зображенні 1.1.10:

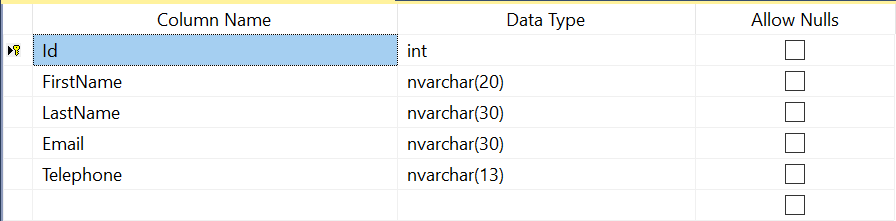
****

**Зображення 1.1.10 Поля таблиці Orders**

* Поле Id є первинним ключем та лічильником замовлень.
* Поле ClientId є зовнішнім ключем до таблиці Clients та містить Id клієнта.
* Поле RegistrationDate містить інформацію про дату та час оформлення замовлення.

**Clients:**

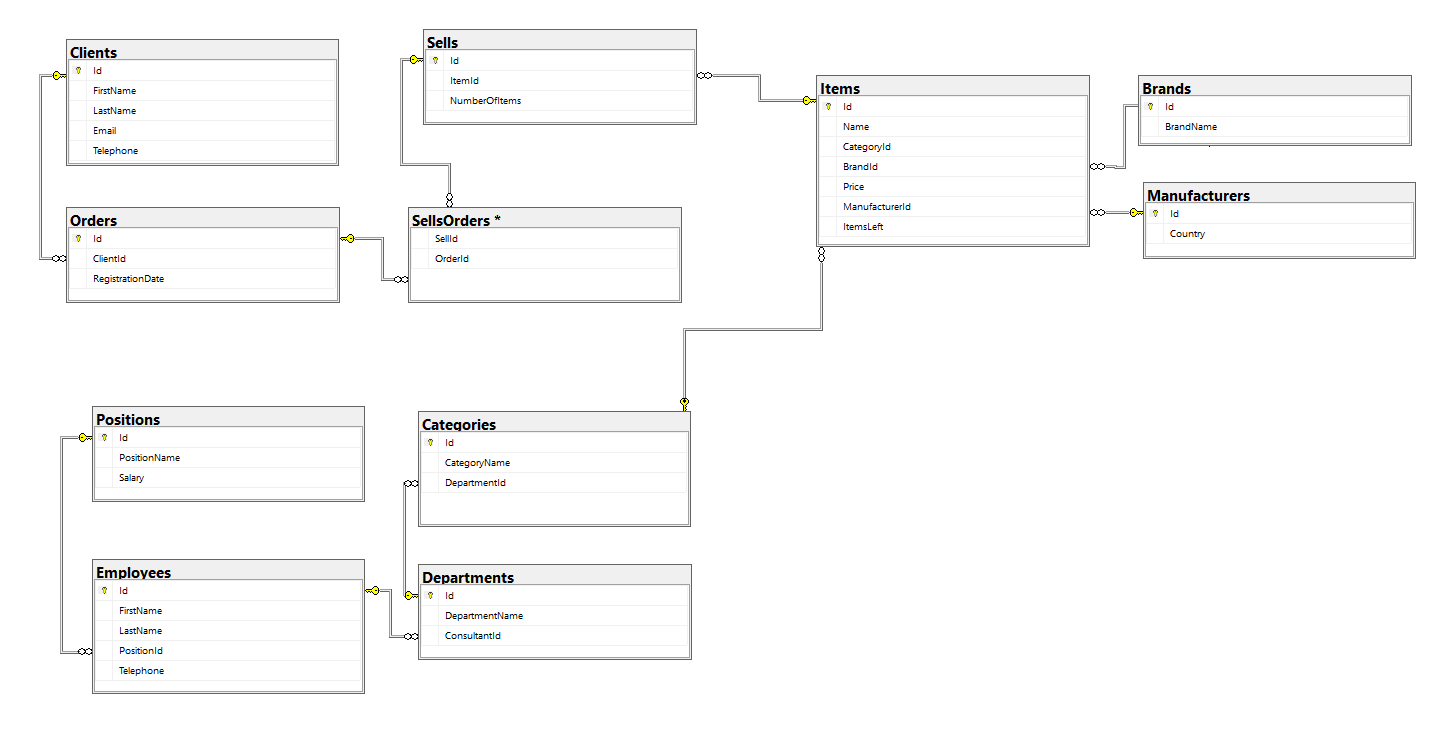
Ця таблиця містить 5 полів, як показано на зображенні 1.1.11:

****

**Зображення 1.1.11 Поля таблиці Clients**

* Поле Id є первинним ключем та лічильником клієнтів магазину.
* Поле FirstName містить ім’я клієнта.
* Поле LastName містить фамілію клієнта.
* Поле Email містить електронну пошту клієнта.
* Поле Telephone містить номер клієнта.

**Схема даних:**

Схема даних – це відображення відносин між таблицями у вигляді діаграми (Зображення 1.2.1).****

**Зображення 1.2.1 Діаграма бази даних**

Є три типа зв’язку між таблицями :

- «один до одного» - для кожного рядку однієї таблиці існує лише один рядок другої таблиці.

- «один до багатьох» - декілька рядків «дочірньої» таблиці зв’язані з одним рядком однієї таблиці

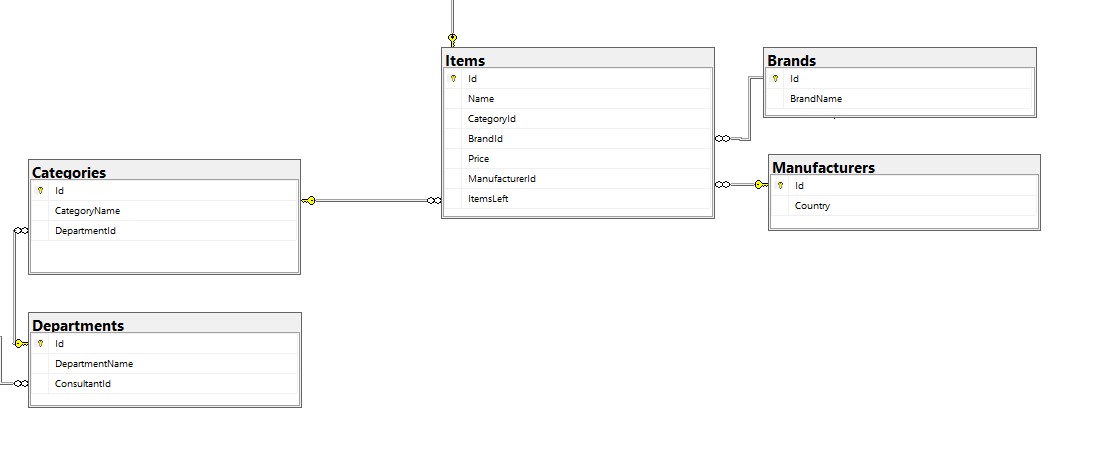
- «багато до багатьох» - декілька рядків однієї таблиці можуть бути зв’язані з одним рядком другої таблиці і навпаки. При такому зв’язку створюється третя таблиця, яка зберігає первинні ключі обох таблиць у вигляді складеного зовнішнього ключів

У моїй діаграмі таблиця Items пов’язана за допомогою зв’язку «один до багатьох» з таблицями (Зображення 1.2.3 ):

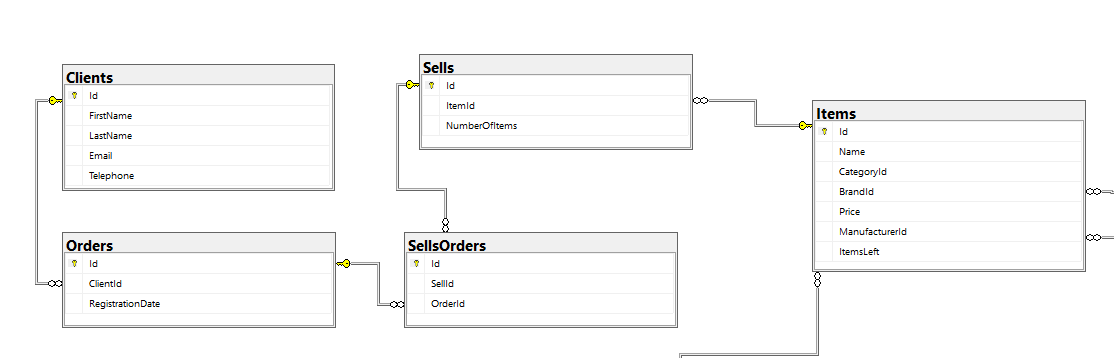
- Brands (за допомогою зовнішнього ключа BrandId )

- Manufacturers (за допомогою зовнішнього ключа ManufacturerId )

- Categories (за допомогою зовнішнього ключа CategoryId )

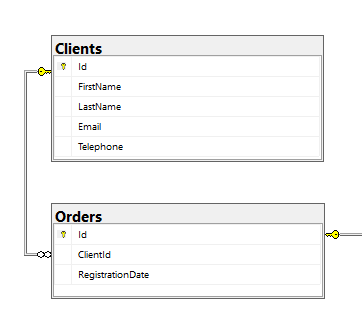
****

**Зображення 1.2.3 Зв’язки таблиці Items**

Таблиця Sells пов’язана з таблицею Items за допомогою зв’язку «один до багатьох»( зовнішній ключ ItemId) (Зображення 1.2.4 ). І пов’язана з таблицею Orders за допомогою зв’язку «багато до багатьох». Для цього була створена таблиця SellsOrders. (Зображення 1.2.4 )****

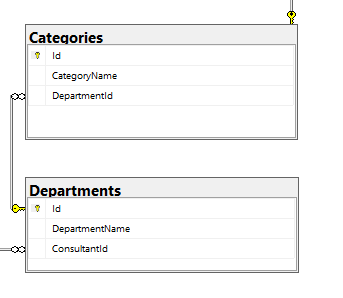
**Зображення 1.2.4 Зв’язки таблиці Sells**

Таблиця Orders пов’язана з таблицею Clients за допомогою зв’язку «один до багатьох»( зовнішній ключ ClientId) (Зображення 15 ).



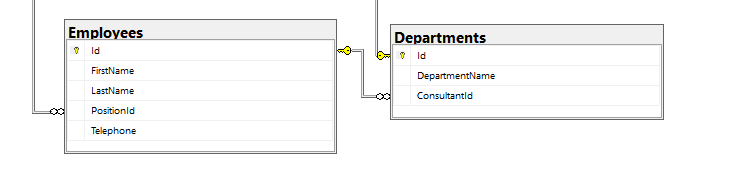
**Зображення 1.2.5 Зв’язки таблиці Orders**

Таблиця Categories пов’язана з таблицею Departments за допомогою зв’язку «один до багатьох»( зовнішній ключ DepartmentId) (Зображення 1.2.6 ).



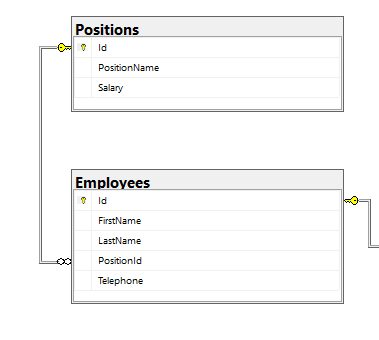
**Зображення 1.2.6 Зв’язки таблиці Categories**

Таблиця Departments пов’язана з таблицею Employees за допомогою зв’язку «один до одного»(зовнішній ключ ConsultantId)(Зображення 1.2.7 ).



**Зображення 1.2.7 Зв’язки таблиці Departments**

Таблиця Employees пов’язана з таблицею Positions за допомогою зв’язку «один до багатьох»( зовнішній ключ PositionId) (Зображення 1.2.8 ).



**Зображення 1.2.8 Зв’язки таблиці Employees**

**Запити**

**Уявлення**

Для бази даних було створено 4 уявлення : AllItems, AllClients, AllOrders, AllWorkers.

**AllItems:**

Цей запит відображає усі товари магазина в базі даних.

Створення запиту:

CREATE VIEW AllItems

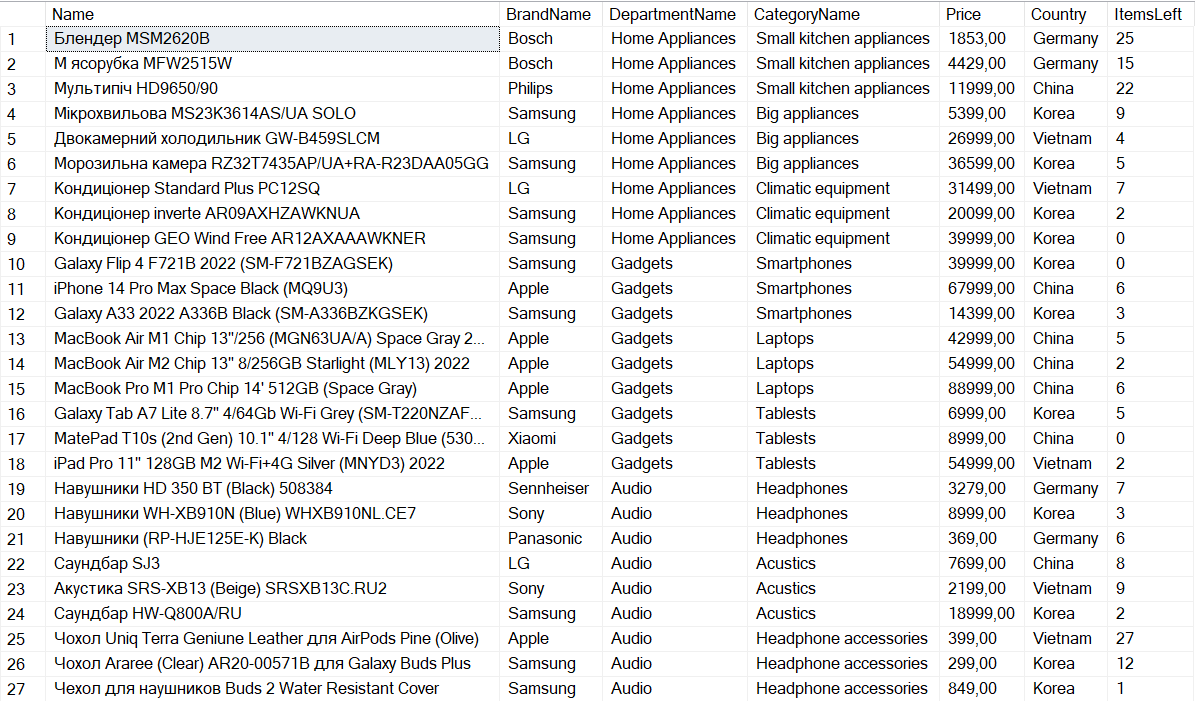
AS

SELECT [Name], BrandName, DepartmentName, CategoryName, Price, Country, ItemsLeft

FROM Items AS I JOIN Brands AS B ON I.BrandId = B.Id JOIN Manufacturers AS M ON I.ManufacturerId = M.Id JOIN Categories AS C ON C.Id = I.CategoryId JOIN Departments AS D ON D.Id = C.DepartmentId;

Запит для виводу інформації:

SELECT \* FROM AllItems;



**Зображення 2.1.1 Вивід всіх товарів у базі даних**

**AllClients:**

Цей запит відображає усіх клієнтів в базі даних.

Створення запиту:

CREATE VIEW AllClients

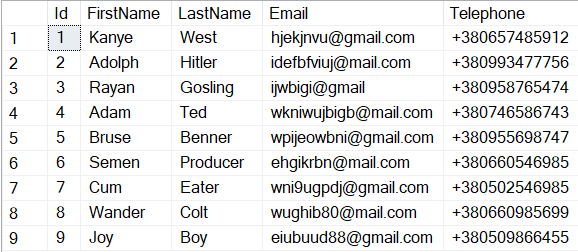
AS

SELECT \*

FROM Clients;

Запит для виводу інформації:

SELECT \* FROM AllClients;



**Зображення 2.1.2 Вивід усіх клієнтів у базі даних**

**AllOrders:**

Цей запит відображає усі замовлення в базі даних.

Створення запиту:

CREATE VIEW AllOrders

AS

SELECT FirstName, LastName, RegistrationDate, [Name], NumberOfItems, NumberOfItems\*Price AS 'Total price'

FROM Clients AS C JOIN Orders AS O ON C.Id = O.ClientId JOIN SellsOrders AS SO ON SO.OrderId = O.Id JOIN Sells AS S ON S.Id = SO.SellId JOIN Items AS I ON I.Id = S.ItemId;

Запит для виводу інформації:

SELECT \* FROM AllOrders



**Зображення 2.1.3 Вивід усіх замовлень у базі даних**

**AllWorkers:**

Цей запит відображає усіх робітників в базі даних.

Створення запиту:

CREATE VIEW AllWorkers

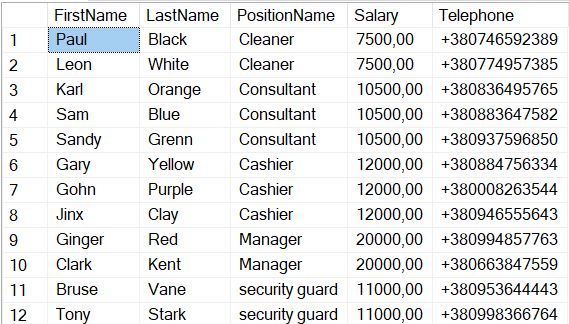
AS

SELECT FirstName, LastName, PositionName, Salary, Telephone

FROM Employees AS E JOIN Positions AS P ON P.Id = E.PositionId

Запит для виводу інформації:

SELECT \* FROM AllWorkers

****

**Зображення 2.1.4 Вивід усіх робітників у базі даних**

**Функції**

Для бази даних було створено 4 функції: SearchBrand, SearchManufacterer, LastMonthSells, LastMonthEncome.

**SearchBrand:**

Ця функція відображає усі товари бренду, який був введений як аргумент функції.

Створення функції:

CREATE FUNCTION SearchBrand(@name nvarchar(50))

RETURNS TABLE

AS

RETURN

(

SELECT BrandName, [Name]

FROM Items AS I JOIN Brands AS B ON B.Id = I.BrandId

WHERE BrandName LIKE @name

);

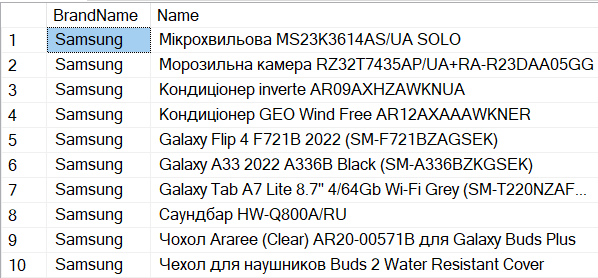
Вивід інформації здійснюється за допомогою передачі бажаного френду до змінної:

DECLARE @name nvarchar(50)

SET @name = 'Samsung'

Виклик функції відбувається наступним чином:

SELECT \* FROM SearchBrand(@name)



**Зображення 2.2.1 Вивід товарів за брендом**

**SearchManufacturer:**

Ця функція відображає усі товари вироблені в країні, яка була введена як аргумент функції.

Створення функції:

CREATE FUNCTION SearchManufacturer(@country nvarchar(50))

RETURNS TABLE

AS

RETURN

(

SELECT Country, [Name]

FROM Items AS I JOIN Manufacturers AS M ON M.Id = I.ManufacturerId

WHERE Country LIKE @country

);

Вивід інформації здійснюється за допомогою передачі бажаної країни до змінної:

DECLARE @country nvarchar(50)

SET @country = 'China'

Виклик функції відбувається наступним чином:

SELECT \* FROM SearchManufacturer(@country)



**Зображення 2.2.2 Вивід товарів за країною виробником**

**LastMonthSells:**

Ця функція відображає усі продажі, що відбулися цього місяця.

Створення функції:

CREATE FUNCTION LastMonthSells()

RETURNS TABLE

AS

RETURN

(

SELECT O.Id, FirstName, LastName, RegistrationDate, I.[Name], NumberOfItems, Price

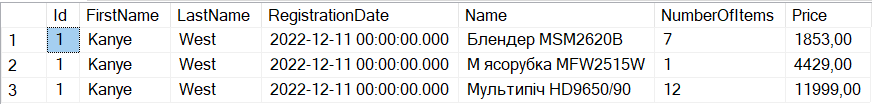
FROM Clients AS C JOIN Orders AS O ON C.Id = O.ClientId JOIN SellsOrders AS SO ON SO.OrderId = O.Id JOIN Sells AS S ON S.Id = So.SellId JOIN Items AS I ON I.Id = S.ItemId

WHERE MONTH(RegistrationDate) = MONTH(GETDATE()) AND YEAR(RegistrationDate) = YEAR(GETDATE())

);

Вивід інформації відбувається наступним чином:

SELECT \* FROM LastMonthSells();



**Зображення 2.2.3 Вивід усіх продажів, що відбулись цього місяця**

**LastMonthEncome:**

Ця функція відображає сумарний прибуток за продажі, що відбулися цього місяця.

Створення функції:

CREATE FUNCTION LastMonthEncome()

RETURNS money

BEGIN

DECLARE @summ money

SELECT @summ = SUM(NumberOfItems \* Price)

FROM Orders AS O JOIN SellsOrders AS SO ON SO.OrderId = O.Id JOIN Sells AS S ON S.Id = SO.SellId JOIN Items AS I ON I.Id= s.ItemId

WHERE MONTH(RegistrationDate) = MONTH(GETDATE()) AND YEAR(RegistrationDate) = YEAR(GETDATE())

RETURN @summ

END

Вивід інформації відбувається наступним чином:

SELECT dbo.LastMonthEncome() AS Total;

****

**Зображення 2.2.4 Вивід прибутку за продажі, що відбулися цього місяця**

**Збережені процедури**

Для бази даних було створено 5 збережених процедур: ItemsCount, OrdersCount, ClientsCount, CancelSell, CancelOrder.

**ItemsCount:**

Ця процедура відображає кількість актуальних пропозицій у магазині.

Створення процедури:

CREATE PROC ItemsCount

AS

DECLARE @togather int

SELECT @togather = COUNT(\*)

FROM Items

WHERE ItemsLeft > 0

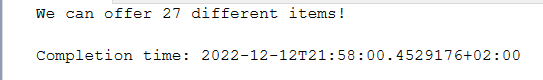
RETURN @togather;

Виклик процедури:

DECLARE @res int

EXEC @res = ItemsCount;

PRINT 'We can offer '+CAST(@res AS nvarchar(10))+' different items!'

****

**Зображення 2.3.1 Вивід кількості актуальних пропозицій**

**OrdersCount:**

Ця процедура відображає кількість зроблених в магазині замовлень.

Створення процедури:

CREATE PROC OrdersCount

AS

DECLARE @togather int

SELECT @togather = COUNT(\*)

FROM Orders

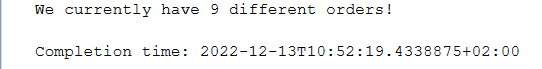
RETURN @togather;

Виклик процедури:

DECLARE @res int

EXEC @res = OrdersCount;

PRINT 'We currently have '+CAST(@res AS nvarchar(10))+' different orders!'

**  
Зображення 2.3.2 Вивід кількості зроблених в магазині замовлень**

**ClientsCount:**

Ця процедура відображає кількість клієнтів магазина.

Створення процедури:

CREATE PROC ClientsCount

AS

DECLARE @togather int

SELECT @togather = COUNT(\*)

FROM Clients

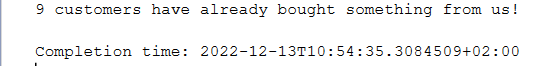
RETURN @togather;

Виклик процедури:

DECLARE @res int

EXEC @res = ClientsCount;

PRINT CAST(@res AS nvarchar(10))+' customers have already bought something from us!'



**Зображення 2.3.3 Вивід кількості клієнтів магазина**

**CancelSell:**

Ця процедура відміняє замовлення певного товару.

Створення процедури:

CREATE PROC CancelSell @sellId int, @orderId int

AS

DELETE FROM SellsOrders

WHERE SellId = @sellId AND @orderId = OrderId

IF NOT EXISTS(SELECT \* FROM SellsOrders WHERE OrderId = @orderId)

BEGIN

DELETE FROM Orders

WHERE Id = @orderId

END

Відміна відбувається після передачі даних про замовлення до змінних:

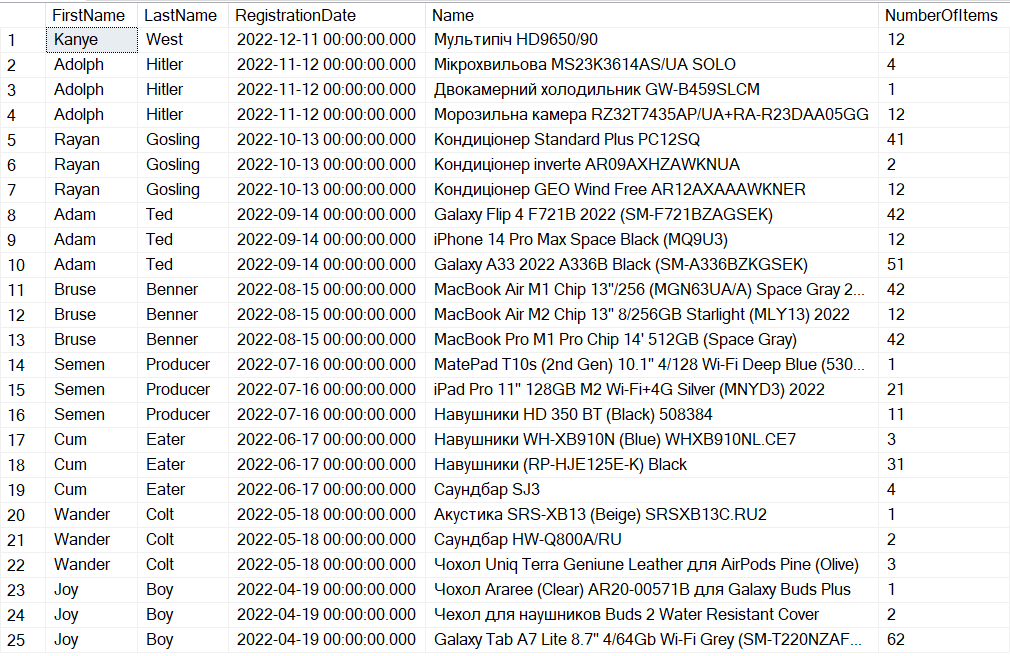
DECLARE @sellId int, @orderId int

SET @sellId = 3

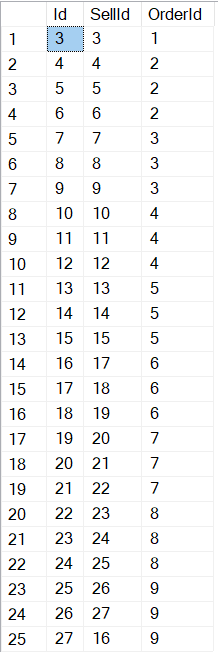
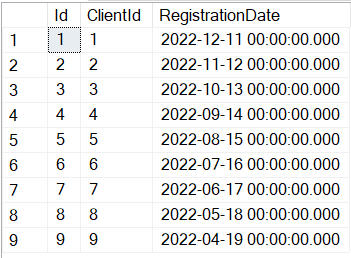
SET @orderId = 1

Виклик процедури:

EXEC CancelSell @sellId, @orderId

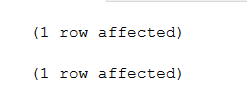


**Зображення 2.3.3.1 Вивід усіх замовлень**



**Зображення 2.3.3.2 Вивід усіх замовлень**

**Зображення 2.3.3.3 Вивід усіх замовлень**

****

**Зображення 2.3.3.4 Результат виконання функції**

**CancelOrder:**

Ця процедура відміняє певне замовлення.

Створення процедури:

CREATE PROC CancelOrder @orderId int

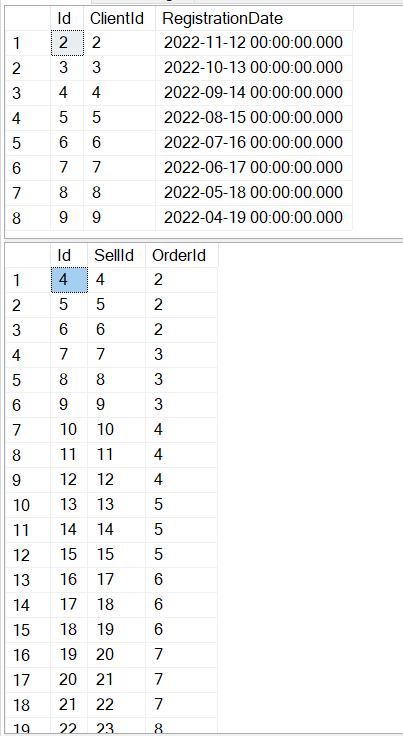
AS

DELETE FROM SellsOrders

WHERE @orderId = OrderId

DELETE FROM Orders

WHERE Id = @orderId

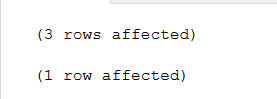
Відміна відбувається після передачі даних про замовлення до змінних:

DECLARE @orderId int

SET @orderId = 2

Виклик процедури:

EXEC CancelSell @orderId

****

**Зображення 2.3.4.2 Результат виконання функції**

**Зображення 2.3.4.1 Вивід усіх замовлень**

**Тригери:**

Для бази даних було створено 3 тригеры: DeleteClient, NewOrderSell, ProperCancellSell.

**DeleteClient:**

Цей тригер був створений, щоб унеможливити видалення клієнтів.

Створення триггера:

CREATE TRIGGER DeleteClient ON Clients

FOR DELETE

AS

PRINT 'You can not delete clients'

ROLLBACK

**NewOrderSell:**

Цей тригер був створений, щоб при додаванні певного товару до замовлення, відбувалася перевірка наявності достатньої кількості товарів, і якщо цієї кількості недостатньо - унеможливити додавання товару до замовлення.

Створення тригеру:

CREATE TRIGGER NewOrderSell ON SellsOrders

FOR INSERT

AS

IF (SELECT ItemsLeft FROM Items AS It JOIN Sells AS S ON S.ItemId = It.Id JOIN inserted AS ins ON ins.SellId = S.Id) < (SELECT NumberOfItems FROM Sells AS S JOIN inserted AS ins ON ins.SellId = S.Id)

BEGIN

PRINT 'Not enough items for this sell'

ROLLBACK

END

ELSE

BEGIN

UPDATE Items

SET ItemsLeft -= (SELECT NumberOfItems FROM inserted AS I JOIN Sells AS S ON S.Id = I.SellId)

WHERE Id = (SELECT I.Id FROM Items AS I JOIN Sells AS S ON S.ItemId = I.Id JOIN inserted AS Ins ON S.Id = Ins.SellId);

END

**ProperCancelSell:**

Цей тригер був створений, щоб при відміні замовлення певного товару, повна кількість товарів повернулася до таблиці Items.

Створення тригеру:

CREATE TRIGGER ProperCancelSell ON SellsOrders

FOR DELETE

AS

UPDATE Items

SET ItemsLeft += (SELECT NumberOfItems FROM deleted AS D JOIN Sells AS S ON S.Id = D.SellId)

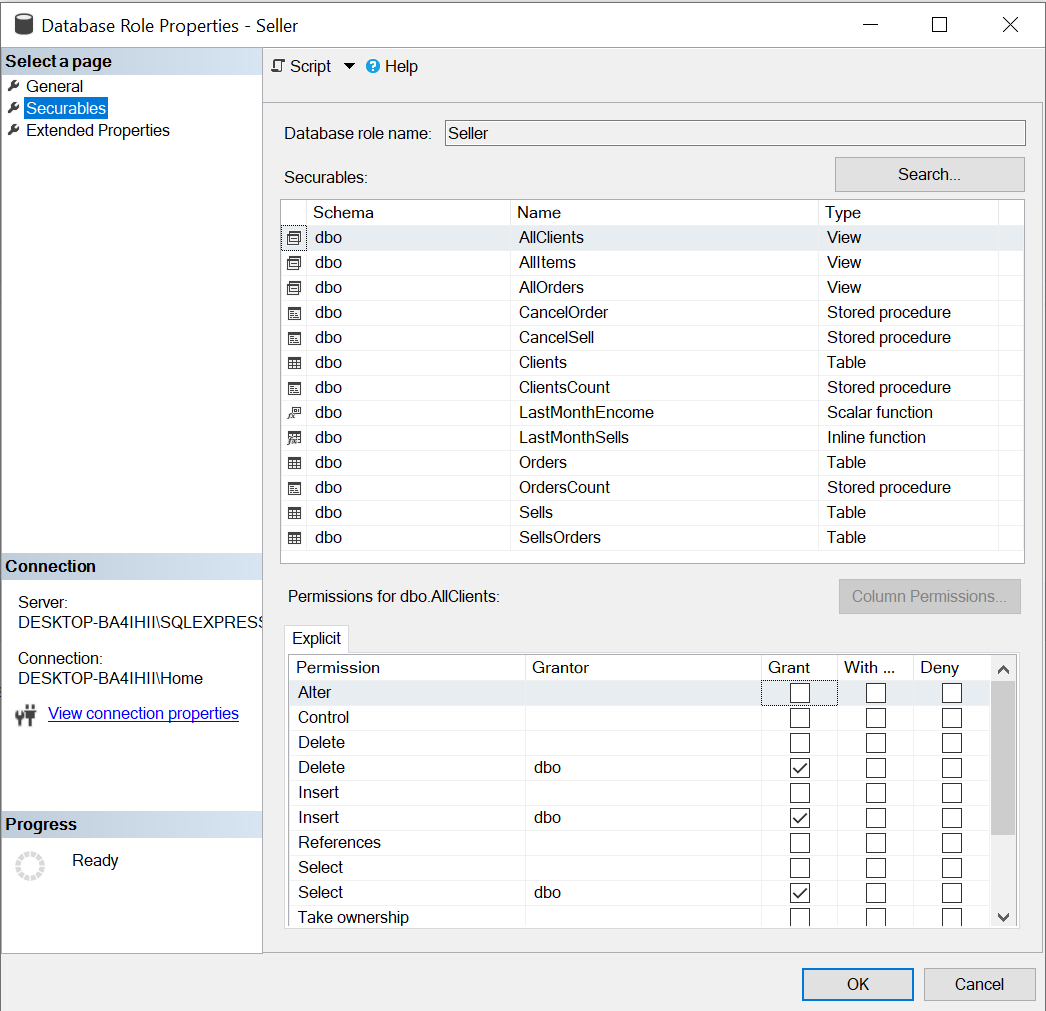
WHERE Id = (SELECT I.Id FROM Items AS I JOIN Sells AS S ON S.ItemId = I.Id JOIN deleted AS D ON S.Id = D.SellId);

**Ролі:**

Для цієї бази даних було створено 4 ролі: Seller, Client, InventoryManager, HR.

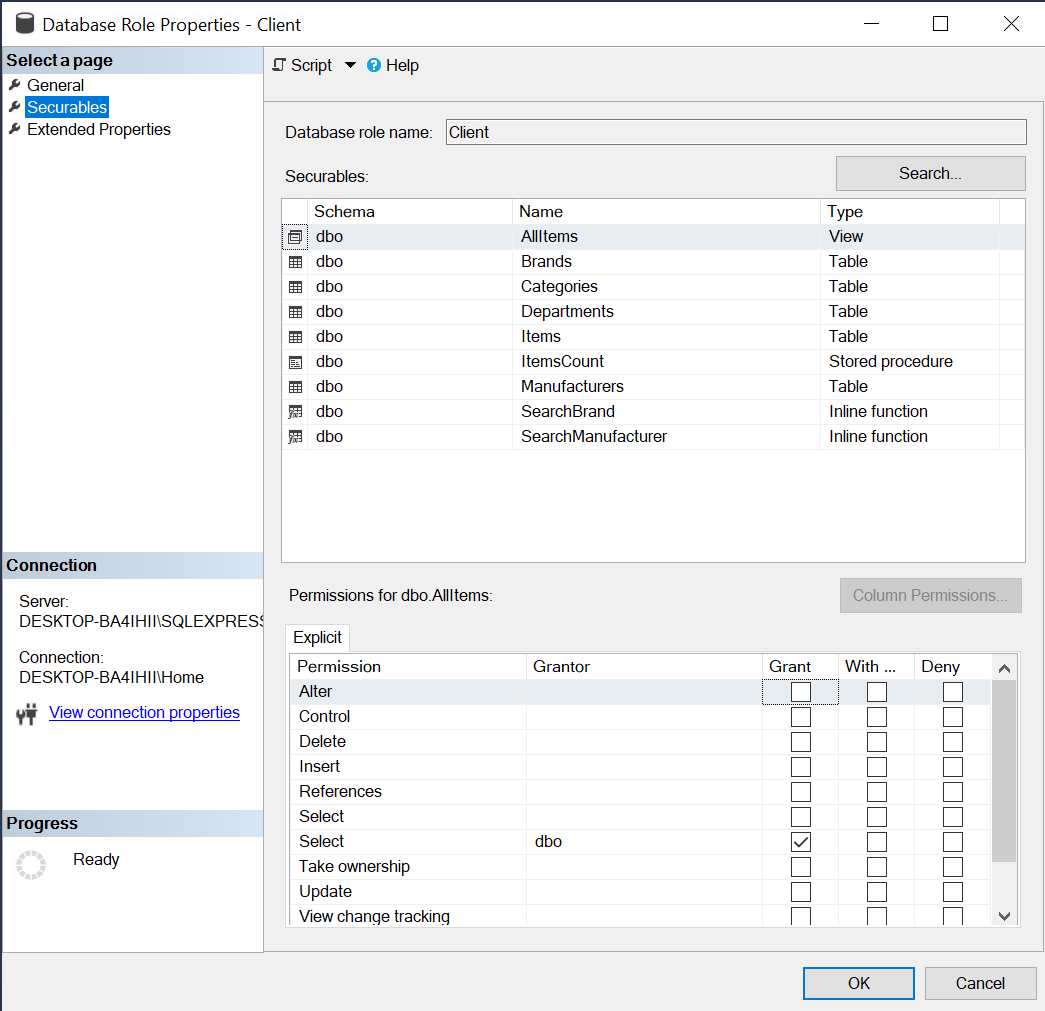
**Seller** -користувачі з цією роллю можуть видаляти, змінювати, додавати та викликати дані з таблиць Clients, SellsOrders, Sells, Orders. А також дивитись уявлення AllItems, AllClients, AllOrders. А також викликати процедуры OrdersCount та ClientsCount. Можуть викликати функції LastMounthSells, LastMounthEncome

( Зображення 3.1).

****

**Зображення 3.1 Роль Seller**

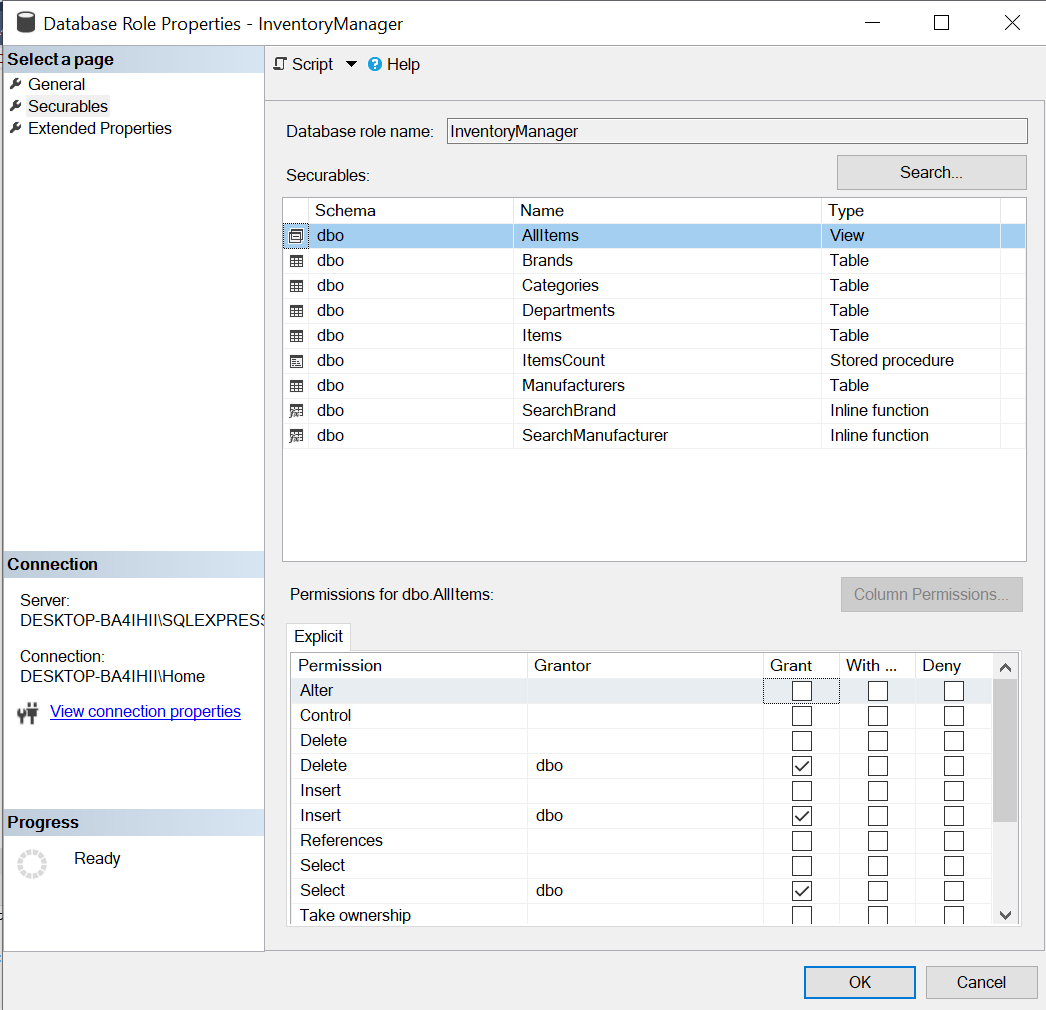
**Client** - користувачі з цією роллю можуть викликати дані з таблиць Items, Brands, Manufacturers, Departments та Categories. Можуть вибирати дані з уявлення AllItems. Можуть викликати функції SearchBarand та SearchManufacturer. Також може викликати процедуру ItemsCount ( Зображення 3.2).

****

**Зображення 3.2 роль Client**

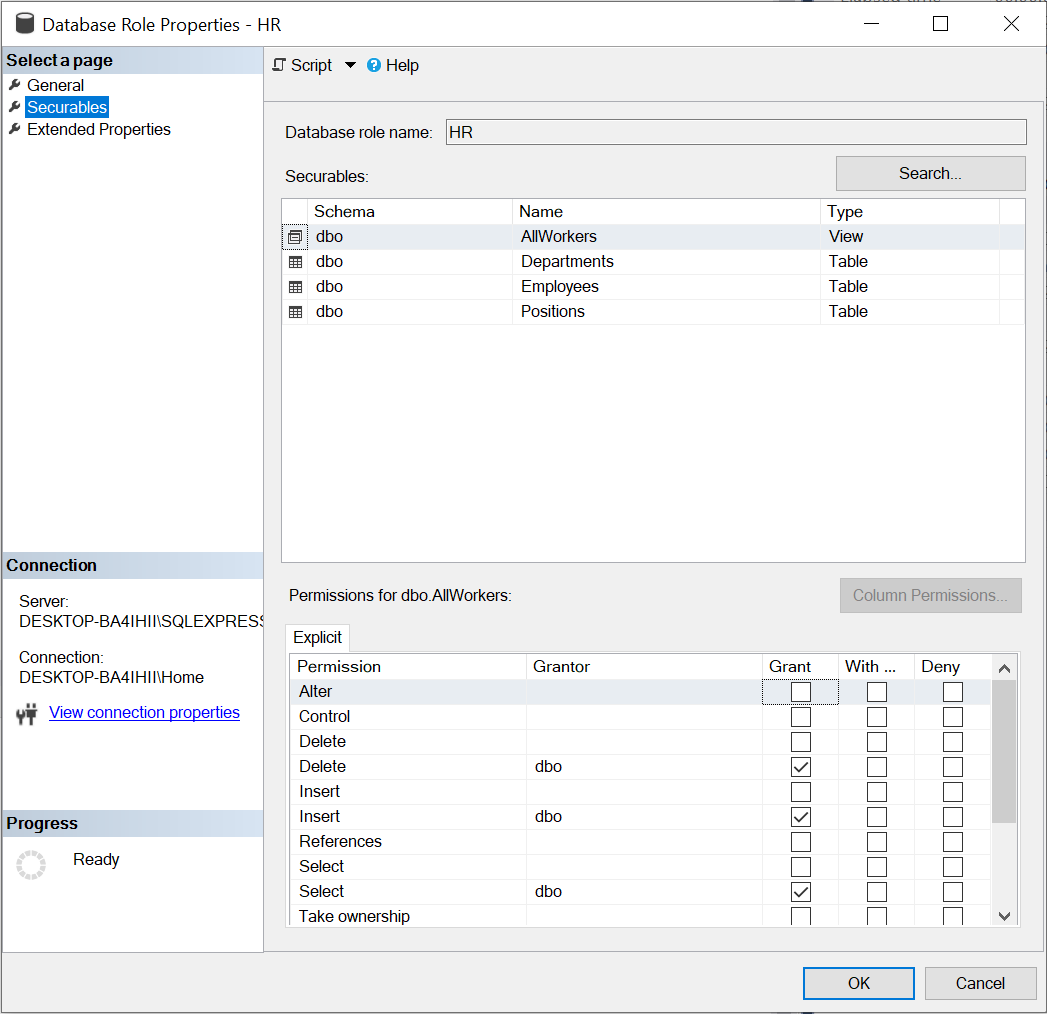
**InventoryManager** - користувачі з цією роллю можуть видаляти, змінювати, оновлювати та вибирати дані з таблиць Brands, Categories, Departments, Items, Manufacturers. Можуть викликати функції SearchBrand, SearchManufacturer. Можуть вибирати дані з уявлення AllItems. Також можуть викликати процедуру ItemsCount

( Зображення 3.3).

****

**Зображення 3.3 Роль InventoryManager**

**HR** - користувачі з цією роллю можуть додавати, змінювати, оновлювати та видаляти дані з таблиць Departments, Employees, Positions. Також можуть вибирати дані з уявлення AllWorkers ( Зображення 3.4)

****

**Зображення 3.4 роль HR**

**Висновок**

У ході виконання технічного завдання було спроектовано базу даних для систематичного зберігання інформації магазину техніки. Ця база даних допомагає систематизувати та швидко обробляти інформацію та зменшує кількість помилок при обробці інформації.

**Використана література**

1. SQL Повна Інструкція. Третє видання. Джеймс Грофф. Пол Вайнберг. Эндрю Оппель. Формат PDF и печать. - : Видавництво "Вильяме" 2015

2. Статті з сайту: https://metanit.com/