BAZA DANYCH SKLEPU Z GRAMI 2021

Jan Skwarczek, Igor Łonak, Paweł Głowacki-Grzyb

Cel główny projektu:

Stworzenie bazy danych służącej do przechowywania informacji dotyczących sklepu internetowego z oddziałami fizycznymi.

Główne cechy projektu:

- Możliwość sprzedaży przedmiotów fizycznych (gier, konsol, ubrań) oraz usług subskrypcyjnych (np. PS+, Xbox Gold)
- System magazynowania przedmiotów
- Funkcjonalność aktywnych zniżek sklepowych i bonusów pensyjnych
- Widoczne oceny danych produktów wystawione przez klientów
- Oznaczenie kategoryjne danych produktów (np. PEGI)
- System logowania pracowników/klientów, obsługa stanowisk

Główne ograniczenia projektu:

- Brak obsługi błędów, transakcji
- Niskie bezpieczeństwo przechowywanych personalnych danych

Strategie pielęgnacji bazy danych:

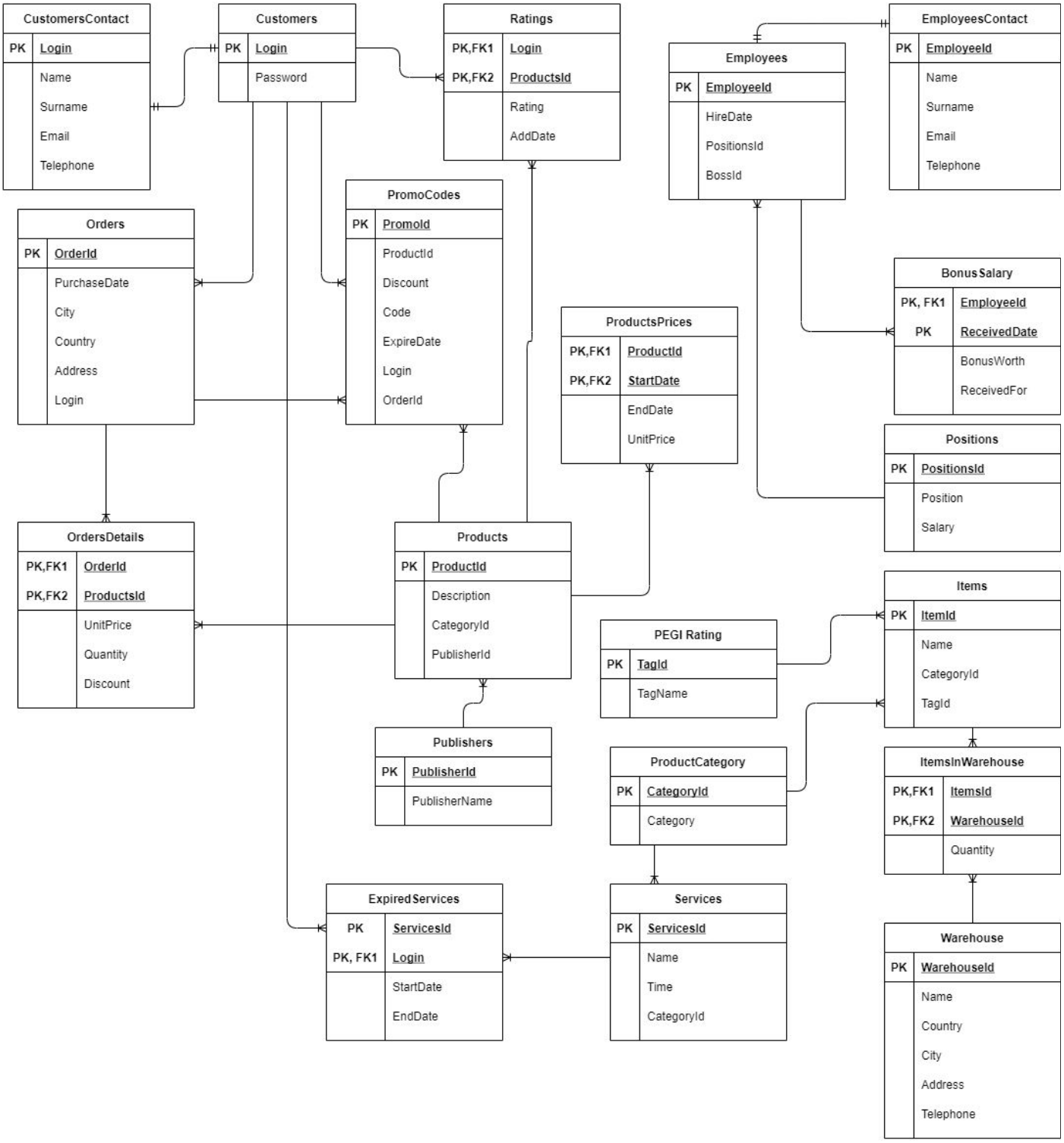
- Tworzenie kopii zapasowej co dwa tygodnie

1. Przedstawienie tabel i ogólnego pomysłu

	TABELA	POLA	CEL
1	BonusSalary	- EmployeeID, PK INT - ReceivedDate, PK DATE - BonusWorth, MONEY - ReceiveFor, NVARCHAR	Informacje o premiach i bonusach pracowników
2	Customers	- Login, PK INT - Password, NVARCHAR	Klienci
3	CustomerContacts	- Login, PK VARCHAR - Name, NVARCHAR - Surname, NVARCHAR - Email, NVARCHAR - Telephone, NVARCHAR	Informacje kontaktowe klientów
4	Employees	- EmployeeId, PK INT - HireDate, DATE - PositionId, INT - BossId, INT	Pracownicy
5	EmployeeContacts	- EmployeeId, PK INT - Name, NVARCHAR - Surname, NVARCHAR - Email, NVARCHAR - Telephone, NVARCHAR	Informacje kontaktowe pracowników
6	ExpiredServices	- ServicesId, PK INT - Login, PK INT - StartDate, DATETIME - EndDate, PK DATETIME	Wygasłe usługi
7	Items	- ItemId, PK INT - Name, NVARCHAR - CategoryId, INT - TagId, INT	Przedmioty sprzedawane w sklepie
8	ItemsInWarehouse	- ItemId, PK INT - WarehouseId, PK INT - Quantity, INT	Informacje o tym, w którym magazynie znajdują się dane przedmioty
9	Orders	- OrderId, PK INT - PurchaseDate, DATETIME - City, NVARCHAR - Country, NVARCHAR - Address, NVARCHAR - Login, INT	Zamówienia złożone w sklepie
10	OrdersDetails	- OrderId, PK INT - ProductId, PK INT - UnitPrice, MONEY - Quantity, SMALLINT - Discount, INT	Szczegółowe informacje o zamówieniach

11	PEGI Rating	- TagID, PK INT - TagName VARCHAR(50)	Kategorie wiekowe
12	Positions	- PositionsId, PK INT - Position, NVARCHAR - Salary, MONEY	Stanowiska pracowników wraz z ich wynagrodzeniem
13	ProductCategory	- CategoryId, PK INT - Category, NVARCHAR	Kategorie produktów
14	Products	- ProductId, PK INT - Description, TEXT - CategoryId, INT - PublisherId, INT	Produkty fizyczne + usługi
15	ProductsPrices	- ProductId, PK INT - StartDate, PK DATETIME - EndDate, DATETIME - UnitPrice, MONEY	Ceny produktów
16	PromoCodes	- PromoId, PK INT - ProductId, INT - Discount, INT - Code, NVARCHAR - ExpireDate, DATETIME - Login, INT - OrderId, INT	Kody promocyjne na dane produkty
17	Publishers	- PublisherId, PK, INT - PublisherName, NVARCHAR	Wydawcy gier
18	Ratings	- Login, PK, INT - ProductsId, PK INT - Rating, SMALLINT - AddDate DATETIME	Oceny produktów
19	Services	- ServiceId, PK INT - Name, NVARCHAR - Time, INT - CategoryId, INT	Usługi subskrypcyjne, które można kupić
20	Warehouses	- WarehouseId, PK, INT - Name, NVARCHAR - Country, NVARCHAR - City, NVARCHAR - Address, NVARCHAR - Telephone, NVARCHAR	Dostępne magazyny

2. Diagram ER


CustomersContact
PK Login
Name
Surname
Email
Telephone
Customers
PK Login
Password
Ratings
PK,FK1 Login
PK,FK2 ProductsId
Rating
AddDate
Employees
PK EmployeeId
HireDate
PositionsId
BossId
EmployeesContact
PK EmployeeId
Name
Surname
Email
Telephone
PromoCodes
PK PromoId
ProductId
Discount
Code
ExpireDate
Login
OrderId
Orders
PK OrderId
PurchaseDate
City
Country
Address
Login
BonusSalary
PK, FK1 EmployeeId
PK ReceivedDate
BonusWorth
ReceivedFor
ProductsPrices
PK,FK1 ProductId
PK,FK2 StartDate
EndDate
UnitPrice
Positions
PK PositionsId
Position
Salary
OrdersDetails
PK,FK1 OrderId
PK,FK2 ProductsId
UnitPrice
Quantity
Discount
Products
PK ProductId
Description
CategoryId
PublisherId
Items
PK ItemId
Name
CategoryId
TagId
PEGI Rating
PK TagId
TagName
Publishers
PK PublisherId
PublisherName
ProductCategory
PK CategoryId
Category
ItemsInWarehouse
PK,FK1 ItemsId
PK,FK2 WarehouseId
Quantity
ExpiredServices
PK ServicesId
PK, FK1 Login
StartDate
EndDate
Services
PK ServicesId
Name
Time
CategoryId
Warehouse
PK WarehouseId
Name
Country
City
Address
Telephone

3. Tworzenie tabel

```
IF OBJECT_ID('dbo.BonusSalary', 'U') IS NOT NULL
        DROP TABLE BonusSalary
GO
CREATE TABLE BonusSalary (
    EmployeeId INT NOT NULL,
    ReceivedDate DATE DEFAULT GETDATE(),
        BonusWorth MONEY NOT NULL,
        ReceiveFor NVARCHAR(30) NOT NULL,
        PRIMARY KEY (EmployeeId,ReceivedDate)
)
GO

IF OBJECT_ID('dbo.Customers', 'U') IS NOT NULL
        DROP TABLE Customers
GO
CREATE TABLE Customers (
    Login  INT NOT NULL,
        Password NVARCHAR(30) NOT NULL,
        PRIMARY KEY (Login)
)
GO

IF OBJECT_ID('dbo.CustomerContacts', 'U') IS NOT NULL
        DROP TABLE CustomerContacts
GO
CREATE TABLE CustomerContacts (
    Login INT NOT NULL,
        Name NVARCHAR(30) NOT NULL,
        Surname NVARCHAR(30) NOT NULL,
        Email NVARCHAR(30) NOT NULL,
        Telephone NVARCHAR(30) NOT NULL,
        PRIMARY KEY (Login)
)
GO

IF OBJECT_ID('dbo.Employees', 'U') IS NOT NULL
        DROP TABLE Employees
GO
CREATE TABLE Employees (
    EmployeeId INT NOT NULL,
        HireDate DATE DEFAULT GETDATE(),
        PositionId INT NOT NULL,
        BossId INT,
        PRIMARY KEY (EmployeeId)
)
GO

IF OBJECT_ID('dbo.EmployeesContacts', 'U') IS NOT NULL
        DROP TABLE EmployeesContacts
GO
CREATE TABLE EmployeesContacts (
    EmployeeId INT NOT NULL,
        Name NVARCHAR(30) NOT NULL,
        Surname NVARCHAR(30) NOT NULL,
        Email NVARCHAR(30) NOT NULL,
        Telephone NVARCHAR(30) NOT NULL,
        PRIMARY KEY (EmployeeId)

)
GO

IF OBJECT_ID('dbo.ExpiredServices', 'U') IS NOT NULL
        DROP TABLE ExpiredServices
```

```
GO
CREATE TABLE ExpiredServices (
    ServiceId INT NOT NULL,
        Login INT NOT NULL,
        StartDate DATETIME NOT NULL,
        EndDate DATETIME DEFAULT GETDATE(),
        PRIMARY KEY (ServiceId,Login,EndDate)

)
GO

IF OBJECT_ID('dbo.Items', 'U') IS NOT NULL
        DROP TABLE Items
GO
CREATE TABLE Items (
    ItemId INT NOT NULL,
        Name NVARCHAR(30) NOT NULL,
        CategoryId INT NOT NULL,
        TagId INT NOT NULL,
        PRIMARY KEY (ItemId)
)
GO

IF OBJECT_ID('dbo.ItemsInWarehouse', 'U') IS NOT NULL
        DROP TABLE ItemsInWarehouse
GO
CREATE TABLE ItemsInWarehouse (
    WarehouseId INT NOT NULL,
        ItemId INT NOT NULL,
        Quantity INT NOT NULL,
        PRIMARY KEY (ItemId, WarehouseId)
)
GO

IF OBJECT_ID('dbo.Orders', 'U') IS NOT NULL
        DROP TABLE Orders
GO
CREATE TABLE Orders (
    OrderId INT NOT NULL,
        PurchaseDate DATETIME DEFAULT GETDATE(),
        City NVARCHAR(30) NOT NULL,
        Country NVARCHAR(30) NOT NULL,
        Address NVARCHAR(30) NOT NULL,
        Login INT NOT NULL,
        PRIMARY KEY (OrderId)
)
GO

IF OBJECT_ID('dbo.OrderDetails', 'U') IS NOT NULL
        DROP TABLE OrderDetails
GO
CREATE TABLE OrderDetails (
        OrderId INT NOT NULL,
        ProductId INT NOT NULL,
        UnitPrice INT NOT NULL,
        Quantity INT NOT NULL,
        Discount INT NOT NULL,
        PRIMARY KEY (OrderId, ProductId)
)
GO

IF OBJECT_ID('dbo.PEGIRating', 'U') IS NOT NULL
        DROP TABLE PEGIRating
GO
CREATE TABLE PEGIRating (
    TagId INT NOT NULL,
        TagName NVARCHAR(30) NOT NULL,
        PRIMARY KEY (TagId)
```

```
)
GO

IF OBJECT_ID('dbo.Positions', 'U') IS NOT NULL
        DROP TABLE Positions
GO
CREATE TABLE Positions (
    PositionId INT NOT NULL,
        Name NVARCHAR(30) NOT NULL,
        Salary MONEY NOT NULL,
        PRIMARY KEY (PositionId)
)
GO

IF OBJECT_ID('dbo.ProductCategory', 'U') IS NOT NULL
        DROP TABLE ProductCategory
GO
CREATE TABLE ProductCategory (
    CategoryId INT NOT NULL,
        Category NVARCHAR(30) NOT NULL,
        PRIMARY KEY (CategoryId)
)
GO

IF OBJECT_ID('dbo.Products', 'U') IS NOT NULL
        DROP TABLE Products
GO
CREATE TABLE Products (
    ProductId INT NOT NULL,
        Description TEXT,
        CategoryId INT NOT NULL,
        PublisherId INT NOT NULL,
        PRIMARY KEY (ProductId)
)
GO

IF OBJECT_ID('dbo.ProductsPrices', 'U') IS NOT NULL
        DROP TABLE ProductsPrices
GO
CREATE TABLE ProductsPrices (
    ProductId INT NOT NULL,
        StartDate DATETIME DEFAULT GETDATE(),
        UnitPrice MONEY NOT NULL,
        PRIMARY KEY (ProductID, StartDate)
)
GO

IF OBJECT_ID('dbo.PromoCodes', 'U') IS NOT NULL
        DROP TABLE PromoCodes
GO
CREATE TABLE PromoCodes (
    PromoId INT NOT NULL,
        ProductId INT NOT NULL,
        Discount INT NOT NULL,
        Code NVARCHAR(30) NOT NULL,
        ExpireDate DATETIME NOT NULL,
        Login INT NOT NULL,
        OrderId INT NOT NULL,
        PRIMARY KEY (PromoId)
)
GO

IF OBJECT_ID('dbo.Publishers', 'U') IS NOT NULL
        DROP TABLE Publishers
GO
CREATE TABLE Publishers (
        PublisherId INT NOT NULL,
    PublisherName VARCHAR(30) NOT NULL,
```

```
        PRIMARY KEY (PublisherId)
)
GO

IF OBJECT_ID('dbo.Ratings', 'U') IS NOT NULL
        DROP TABLE Ratings
GO
CREATE TABLE Ratings (
    Login INT NOT NULL,
        ProductId INT NOT NULL,
        Rating SMALLINT NOT NULL,
        AddDate DATETIME DEFAULT GETDATE(),
        PRIMARY KEY (Login, ProductId)
)
GO

IF OBJECT_ID('dbo.Services', 'U') IS NOT NULL
        DROP TABLE Services
GO
CREATE TABLE Services (
    ServiceId INT NOT NULL,
        Name NVARCHAR(30) NOT NULL,
        Time INT NOT NULL,
        CategoryId INT NOT NULL,
        PRIMARY KEY (ServiceId)
)
GO

IF OBJECT_ID('dbo.Warehouses', 'U') IS NOT NULL
        DROP TABLE Warehouses
GO
CREATE TABLE Warehouses (
    WarehouseId INT NOT NULL,
        Name NVARCHAR(30) NOT NULL,
        Country NVARCHAR(30) NOT NULL,
        City NVARCHAR(30) NOT NULL,
        Address NVARCHAR(30) NOT NULL,
        Telephone NVARCHAR(30) NOT NULL,
        PRIMARY KEY (WarehouseId)
)
GO
```

4. Klucze obce

```
ALTER TABLE BonusSalary
        ADD FOREIGN KEY (EmployeeId)
                REFERENCES Employees (EmployeeId)
                ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE CustomerContacts
        ADD FOREIGN KEY (Login)
                REFERENCES Customers (Login)
                ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE EmployeesContacts
        ADD FOREIGN KEY (EmployeeId)
                REFERENCES Employees (EmployeeId)
                ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE ExpiredServices
        ADD FOREIGN KEY (ServiceId)
                REFERENCES Services (ServiceId)
GO

ALTER TABLE Items
        ADD FOREIGN KEY (ItemId)
        REFERENCES Products (ProductId)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE ItemsInWarehouse
        ADD FOREIGN KEY (WarehouseId)
        REFERENCES Warehouses(WarehouseId)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE ItemsInWarehouse
        ADD FOREIGN KEY (ItemId)
        REFERENCES Items (ItemId)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE OrderDetails
        ADD FOREIGN KEY (OrderId)
        REFERENCES Orders (OrderId)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE ProductsPrices
        ADD FOREIGN KEY (ProductId)
        REFERENCES Products (ProductId)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE PromoCodes
        ADD FOREIGN KEY (ProductId)
```

```
        REFERENCES Products (ProductId)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE Ratings
        ADD FOREIGN KEY (Login)
        REFERENCES Customers (Login)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO

ALTER TABLE Ratings
        ADD FOREIGN KEY (ProductId)
        REFERENCES Products(ProductId)
        ON DELETE CASCADE
        ON UPDATE CASCADE
GO
```

5. Sekwencje

```
IF OBJECT_ID('ProductSequence', 'SO') IS NOT NULL
DROP SEQUENCE ProductSequence
GO

CREATE SEQUENCE ProductSequence
    START WITH 1
    INCREMENT BY 1
GO

IF OBJECT_ID('EmployeeSequence', 'SO') IS NOT NULL
DROP SEQUENCE EmployeeSequence
GO

CREATE SEQUENCE EmployeeSequence
    START WITH 1
    INCREMENT BY 1
GO

IF OBJECT_ID('CustomerSequence', 'SO') IS NOT NULL
DROP SEQUENCE CustomerSequence
GO

CREATE SEQUENCE CustomerSequence
    START WITH 1
    INCREMENT BY 1
GO
```

6. Funkcje

```
IF OBJECT_ID ('MostExpensiveInCategory','TF') IS NOT NULL
DROP FUNCTION MostExpensiveInCategory
GO

CREATE FUNCTION MostExpensiveInCategory (@Category NVARCHAR(30))
RETURNS @Output TABLE (ProductId INT,UnitPrice MONEY)
AS
BEGIN
            INSERT INTO @Output
            SELECT TOP 1 A.ProductId, UnitPrice FROM Products
A,ProductsPrices B,ProductCategory C
            WHERE A.ProductId=B.ProductId AND A.CategoryId=C.CategoryId AND
C.Category LIKE @Category
            ORDER BY UnitPrice DESC
            RETURN
END
GO
```

MostExpensiveInCategory zwraca najdroższy produkt w kategorii zadanej argumentem.

```
IF OBJECT_ID ('EmployeeSince','TF') IS NOT NULL
DROP FUNCTION EmployeeSince
GO

CREATE FUNCTION EmployeeSince (@StartDate DATETIME)
RETURNS @Output TABLE (Name NVARCHAR(30), HireDate DATETIME)
AS
BEGIN
      INSERT INTO @Output
      SELECT Name, HireDate FROM Employees A,EmployeesContacts B
      WHERE A.EmployeeId=B.EmployeeId AND A.HireDate>=@StartDate
      ORDER BY HireDate DESC
      RETURN
END
GO
```

EmployeeSince zwraca tabelę pracowników, którzy zostali zatrudnieni po dacie zadanej argumentem.

```sql
IF OBJECT_ID ('WarehouseItems','TF') IS NOT NULL
DROP FUNCTION WarehouseItems
GO

CREATE FUNCTION WarehouseItems (@Item NVARCHAR(30))
RETURNS @Output TABLE (WarehouseName NVARCHAR(30), Quantity INT)
AS
BEGIN
	INSERT INTO @Output
	SELECT A.Name, Quantity FROM Warehouses A, ItemsInWarehouse B,Items C
	WHERE A.WarehouseId=B.WarehouseId AND B.ItemId=C.ItemId AND C.Name=@Item
	ORDER BY Quantity DESC
	RETURN
END
GO
```

WarehouseItems zwraca tabelę magazynów, w których znajduje się przedmiot zadany argumentem oraz ilość tego przedmiotu w magazynie.

```sql
IF OBJECT_ID ('EmployeeOnPosition','FN') IS NOT NULL
DROP FUNCTION EmployeeOnPosition
GO

CREATE FUNCTION EmployeeOnPosition (@Position NVARCHAR(30))
RETURNS  INT
AS
BEGIN
		RETURN(SELECT COUNT(EmployeeId) FROM Employees A,Positions B
		WHERE A.PositionId=B.PositionId AND B.Name=@Position)
END
GO
```

EmployeeOnPosition zwraca ilość pracowników na danym stanowisku.

```sql
IF OBJECT_ID ('DateIncome','FN') IS NOT NULL
DROP FUNCTION DateIncome
GO

CREATE FUNCTION DateIncome (@Date DATE)
RETURNS MONEY
AS
BEGIN
		RETURN(SELECT SUM(A.Quantity*A.UnitPrice) FROM OrderDetails A, Orders
B
		WHERE A.OrderId=B.OrderId AND CONVERT(date,B.PurchaseDate) = @Date)
END
GO
```

DateIncome zwraca zysk ze sprzedaży z dnia zadanego parametrem.

7. Widoki

```
IF OBJECT_ID ('EmployeeNumber','v') IS NOT NULL
DROP VIEW EmployeeNumber
GO

CREATE VIEW EmployeeNumber
AS
          SELECT COUNT(EmployeeId) Number FROM Employees
GO
```

EmployeeNumber podgląda ilość pracowników po *ID*.

```
IF OBJECT_ID ('CustomerNumber','v') IS NOT NULL
DROP VIEW CustomerNumber
GO

CREATE VIEW CustomerNumber
AS
          SELECT COUNT(Login) Number FROM Customers
GO
```

CustomerNumber podgląda ilość klientów po *Loginie*.

```
IF OBJECT_ID ('TOP5Category','v') IS NOT NULL
DROP VIEW TOP5Category
GO

CREATE VIEW TOP5Category
AS
          SELECT TOP 5 D.Category FROM
          (SELECT SUM(B.Quantity*B.UnitPrice) Amount, A.Category
FROM ProductCategory A,OrderDetails B, Products C
          WHERE A.CategoryId=C.CategoryId AND
B.ProductId=C.ProductId
          GROUP BY A.Category) D
          ORDER BY D.Amount
GO
```

TOP5Category podgląda pięć najlepiej “sprzedających się” kategorii produktów.

8. Procedury

```
IF OBJECT_ID('AddItem','p') IS NOT NULL
DROP PROCEDURE Additem
GO

CREATE PROCEDURE AddItem (@Name NVARCHAR(30), @Category NVARCHAR(30), @Description TEXT, @Tag
NVARCHAR(30),@PublisherName NVARCHAR(30),
                                                    @UnitPrice MONEY, @StartDate DATETIME = NULL)
AS
                                IF ((SELECT TOP 1 A.Name FROM Items A WHERE @Name LIKE A.Name) IS NULL
                                AND (SELECT TOP 1 A.Category FROM ProductCategory A WHERE @Category
LIKE A.Category)IS NOT NULL
                                AND (SELECT TOP 1 A.TagName FROM PEGIRating A WHERE A.TagName LIKE
@Tag) IS NOT NULL)
                                BEGIN
                                                DECLARE @CategoryId INT
                                                DECLARE @ItemId INT
                                                DECLARE @TagId INT
                                                DECLARE @PublisherId INT
                                                IF @StartDate IS NULL
                                                SET @StartDate = GETDATE()
                                                SET @ItemId = NEXT VALUE FOR ProductSequence
                                                SET @CategoryId = (SELECT TOP 1 A.CategoryId FROM
ProductCategory A WHERE A.Category LIKE @Category)
                                                SET @TagId = (SELECT TOP 1 A.TagId FROM PEGIRating A
WHERE A.TagName LIKE @Tag)
                                                SET @PublisherId = (SELECT TOP 1 A.PublisherId FROM
Publishers A WHERE A.PublisherName LIKE @PublisherName)
                                                INSERT INTO Products
                                                VALUES (@ItemId, @Description, @CategoryId,
@PublisherId)
                                                INSERT INTO Items
                                                VALUES (@ItemId, @Name, @CategoryId, @TagId)
                                                INSERT INTO ProductsPrices
                                                VALUES (@ItemId, @StartDate, @UnitPrice)

                                END

GO
```

AddItem dodaje dany przedmiot do tabel ***Items***, ***Products*** i ***ProductsPrices***.

```
IF OBJECT_ID('AddEmployee','p') IS NOT NULL
DROP PROCEDURE AddEmployee
GO

CREATE PROCEDURE AddEmployee (@Name NVARCHAR(30), @Surname NVARCHAR(30), @Email NVARCHAR(30), @Position
NVARCHAR(30),
                                                    @Telephone NVARCHAR(30), @HireDate
DATETIME = NULL, @BossId INT = NULL)
AS
                              IF (SELECT TOP 1 A.Name FROM Positions A WHERE A.Name LIKE @Position)
IS NOT NULL
                              BEGIN
                                        DECLARE @PositionId INT
                                        DECLARE @EmployeeId INT
                                        IF @HireDate IS NULL
                                        SET @HireDate = GETDATE()
                                        SET @PositionId = (SELECT TOP 1 A.PositionId FROM
Positions A WHERE A.Name LIKE @Position)
                                        SET @EmployeeId = NEXT VALUE FOR EmployeeSequence
                                        INSERT INTO Employees
                                        VALUES (@EmployeeId,@HireDate, @PositionId, @BossId)
                                        INSERT INTO EmployeesContacts
                                        VALUES (@EmployeeId,@Name, @Surname, @Email,
@Telephone)
                              END

GO
```

AddEmployee dodaje nowego pracownika do tabel ***Employees*** i ***EmployeeContacts***.

```
IF OBJECT_ID('AddCustomer','p') IS NOT NULL
DROP PROCEDURE AddCustomer
GO

CREATE PROCEDURE AddCustomer (@Password NVARCHAR(30), @Name NVARCHAR(30), @Surname
NVARCHAR(30), @Email NVARCHAR(30), @Telephone NVARCHAR(30))
AS
                              BEGIN
                                        DECLARE @CustomerId INT
                                        SET @CustomerId = NEXT VALUE FOR
CustomerSequence
                                        INSERT INTO Customers
                                        VALUES (@CustomerId, @Password)
                                        INSERT INTO CustomerContacts
                                        VALUES (@CustomerId, @Name, @Surname,
@Email, @Telephone)
                              END

GO
```

AddCustomer dodaje nowego klienta do tabel ***Customers*** i ***CustomerContacts***.

```
IF OBJECT_ID('AddService','p') IS NOT NULL
DROP PROCEDURE AddService
GO

CREATE PROCEDURE AddService (@Name NVARCHAR(30), @Description TEXT, @Time INT, @Category NVARCHAR(30),
@PublisherName NVARCHAR(30),
                                                        @UnitPrice MONEY, @StartDate DATETIME= NULL)
AS
                                IF ((SELECT TOP 1 A.Name FROM Items A WHERE @Name LIKE A.Name) IS NULL
                                AND (SELECT TOP 1 A.Category FROM ProductCategory A WHERE @Category
LIKE A.Category)IS NOT NULL)
                                BEGIN
                                                DECLARE @ServiceId INT
                                                DECLARE @CategoryId INT
                                                DECLARE @PublisherId INT
                                                IF @StartDate IS NULL
                                                SET @StartDate = GETDATE()
                                                SET @ServiceId = NEXT VALUE FOR ProductSequence
                                                SET @CategoryId = (SELECT TOP 1 A.CategoryId FROM
ProductCategory A WHERE A.Category LIKE @Category)
                                                SET @PublisherId = (SELECT TOP 1 A.PublisherId FROM
Publishers A WHERE A.PublisherName LIKE @PublisherName)
                                                INSERT INTO Products
                                                VALUES (@ServiceId, @Description, @CategoryId,
@PublisherId)
                                                INSERT INTO Services
                                                VALUES (@ServiceId, @Name, @Time, @CategoryId)
                                                INSERT INTO ProductsPrices
                                                VALUES (@ServiceId, @StartDate, @UnitPrice)
                                END
GO
```

AddService dodaje nową usługę do tabel *Products*, *Services* i *ProductsPrices*.

```
CREATE PROCEDURE AddItemsToWarehouse (@WarehouseName NVARCHAR(30), @ItemName NVARCHAR(30), @Quantity
INT)
AS
                                IF ((SELECT TOP 1 A.Name FROM Items A WHERE @ItemName LIKE A.Name) IS
NOT NULL
                                AND (SELECT TOP 1 A.Name FROM Warehouses A WHERE @WarehouseName LIKE
A.Name)IS NOT NULL)
                                BEGIN
                                                DECLARE @WarehouseId INT
                                                DECLARE @ItemId INT
                                                SET @WarehouseId = (SELECT TOP 1 A.WarehouseId FROM
Warehouses A WHERE @WarehouseName LIKE A.Name)
                                                SET @ItemId = (SELECT TOP 1 A.ItemId FROM Items A
WHERE @ItemName LIKE A.Name)
                                                IF (SELECT A.Quantity FROM ItemsInWarehouse A WHERE
A.WarehouseId=@WarehouseId and A.ItemId=@ItemId) IS NOT NULL
                                                BEGIN
                                                                UPDATE ItemsInWarehouse
                                                                SET Quantity = Quantity + @Quantity
                                                                WHERE WarehouseId=@WarehouseId and
ItemId=@ItemId
                                                END
                                                ELSE
                                                BEGIN
                                                                INSERT INTO ItemsInWarehouse
                                                                VALUES (@WarehouseId, @ItemId, @Quantity)
                                                END
                                END

GO
```

AddItemsToWarehouse dodaje nowe przedmioty do tabel *ItemsInWarehouse*.

9. Wyzwalacze

```
IF OBJECT_ID ('DeleteItem', 'TR') IS NOT NULL
   DROP TRIGGER DeleteItem;
GO

CREATE TRIGGER DeleteItem ON Items
AFTER DELETE
AS
      DELETE FROM Products
      WHERE ProductId IN (SELECT ItemId FROM DELETED)
      DELETE FROM ItemsInWarehouses
      WHERE ProductId IN (SELECT ItemId FROM DELETED)
GO
```

DeleteItem po usunięciu przedmiotu z tabeli *Items* usuwa go również z tabel *Products* oraz *ItemsInWarehouses*.

```
IF OBJECT_ID ('DeleteService', 'TR') IS NOT NULL
   DROP TRIGGER DeleteService;
GO

CREATE TRIGGER DeleteService ON Services
AFTER DELETE
AS
      DELETE FROM Products
      WHERE ProductId IN (SELECT ServiceId FROM DELETED)
GO
```

DeleteService po usunięciu usługi z tabeli *Services* usuwa ją również z tabeli *Products*.

```
IF OBJECT_ID ('DeleteCustomer', 'TR') IS NOT NULL
   DROP TRIGGER DeleteCustomer;
GO

CREATE TRIGGER DeleteCustomer ON Customers
INSTEAD OF DELETE
AS
      DELETE FROM CustomerConstacts
      WHERE Login IN (SELECT Login FROM DELETED)
      DELETE FROM Customers
      WHERE Login IN (SELECT Login FROM DELETED)
GO
```

DeleteCustomer zamiast usunięcia klienta jedynie z tabeli *Customers* usuwa go również z tabeli *CustomerContacts*.

```
IF OBJECT_ID ('DeleteEmployee', 'TR') IS NOT NULL
   DROP TRIGGER DeleteEmployee;
GO

CREATE TRIGGER DeleteEmployee ON Employees
INSTEAD OF DELETE
AS
      DELETE FROM EmployeesContacts
      WHERE EmployeeId IN (SELECT EmployeeId FROM DELETED)
      DELETE FROM Employees
      WHERE EmployeeId IN (SELECT EmployeeId FROM DELETED)
GO
```

DeleteEmployee zamiast usunięcia pracownika jedynie z tabeli ***Employees*** usuwa go również z tabeli ***EmployeeContacts***.

```
IF OBJECT_ID ('DeleteWarehouse', 'TR') IS NOT NULL
   DROP TRIGGER DeleteWarehouse;
GO

CREATE TRIGGER DeleteWarehouse ON Warehouses
INSTEAD OF DELETE
AS
      DELETE FROM ItemsInWarehouse
      WHERE WarehouseId IN (SELECT WarehouseId FROM DELETED)
      DELETE FROM Warehouses
      WHERE WarehouseId IN (SELECT WarehouseId FROM DELETED)
GO
```

DeleteWarehouse zamiast usunięcia magazynu jedynie z tabeli ***Warehouses*** usuwa go również z tabeli ***ItemsInWarehouses***.

10. Przykładowe zapytania

```
EXEC AddCustomer 'abcd1234','Kur', 'Bankiwa', 'kur.bankiwa@gmail.com', '569345333'
SELECT * FROM Customers A, CustomerContacts B WHERE A.Login = B.Login

INSERT INTO Positions
VALUES (1,'szef totalny',20000)
EXEC AddEmployee 'Xorest', 'Możejko', 'xorest.mozejko@o2.pl', 'szef
totalny','123654738'
SELECT * FROM Employees A, EmployeesContacts B WHERE A.EmployeeId = B.EmployeeId

INSERT INTO ProductCategory
VALUES (1, 'FPS')
INSERT INTO PEGIRating
VALUES (1, '18')
INSERT INTO Publishers
VALUES (1, 'Activision')
EXEC AddItem 'Quake 3', 'FPS', '1999 multiplayer-focused first-person shooter
developed by id Software', '18','Activision', 15.00
SELECT * FROM Items A, Products B WHERE A.ItemId = B.ProductId

INSERT INTO Warehouses
VALUES (1, 'Sakura', 'Japonia', 'Kyoto', 'Shichijo-dori 113', '+81 75 353 4126')
EXEC AddItemsToWarehouse 'Sakura', 'Quake 3', 10
SELECT * FROM Warehouses A, ItemsInWarehouse B WHERE A.WarehouseId= B.WarehouseId

INSERT INTO ProductCategory
VALUES (2, 'Premium time')
INSERT INTO Publishers
VALUES (2, 'Discord')
EXEC AddService 'Discord Nitro', 'Discord premium time', 31, 'Premium time',
'Discord', 10.00
SELECT * FROM Products A, Services B WHERE A.ProductId=B.ServiceId

SELECT * FROM EmployeeSince(2021-02-19)

SELECT * FROM MostExpensiveInCategory('Premium time')
SELECT * FROM MostExpensiveInCategory('FPS')

SELECT * FROM WarehouseItems('Quake 3')

print dbo.EmployeeOnPosition('szef totalny')

INSERT INTO Orders
VALUES (1,default,'Kyoto','Japonia','Shichijo-dori 112',1)
INSERT INTO OrderDetails
VALUES (1, 3, 15.00, 5, 0)
print dbo.DateIncome('2021-02-19')
SELECT * FROM Orders A, OrderDetails B WHERE A.OrderId=B.OrderID
```