PROJEKT I WYKONANIE BAZY DANYCH DLA SKLEPU RTV

Michał Sałek | Grupa: lab11/1/ISN | Indeks: 13809

Opis problemu:

Założeniem naszej bazy danych jest przechowywanie danych w tabakach o klientach, pracownikach, produktach, zamówieniach i fakturach. Każda tabela zwiera dane na ich temat. System bazodanowy pozwala na zarządzanie danymi i zdecydowanie szybsze ich wyszukiwanie i zorganizowanie. Baza tworzona jest w języku SQL i działać będzie na Microsoft SQL Server. Baza wpierać będzie stronę internatową sklepu.

Baza pozwala kontrolować zamówienia, zatrudnionych pracowników, klientów, produkty oraz ich kategorie. Dzięki bazie danych będzie można zacząć korzystać z faktur elektronicznych.

Baza danych umożliwia:

- wprowadzanie, usuwanie i aktualizacje danych o klientach

- wprowadzanie, usuwanie i aktualizacje danych o pracownikach

- przyznawanie premii pracownikom (standardowo 200)

- wyszukiwanie zamówień

- wprowadzanie produktów do oferty

- sortowanie produktów względem kategorii

- sprawdzanie sprzedaży względem kategorii oraz producentów

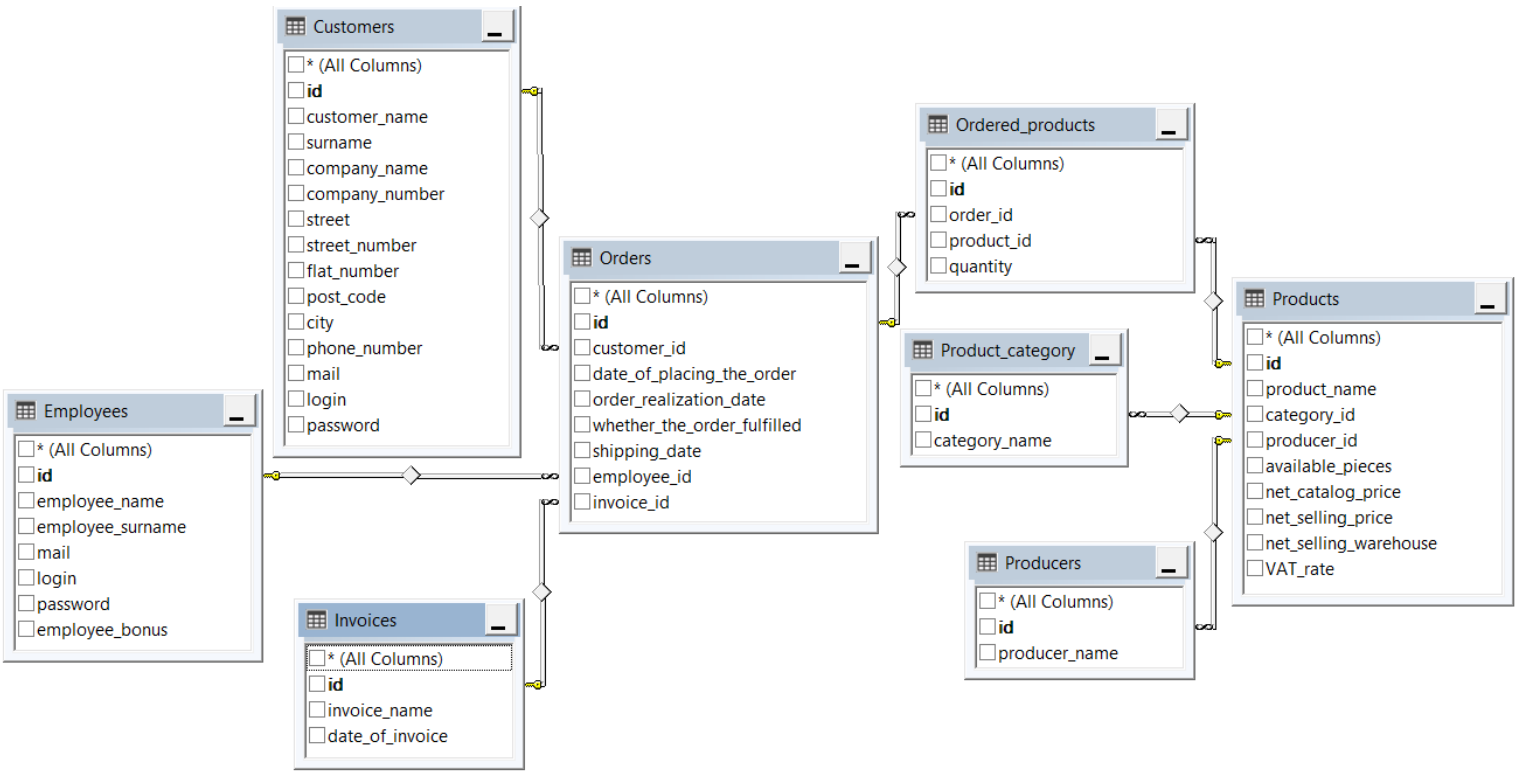
- wystawianie faktur

Projekt podzielny jest na segmenty. Znajduje się nim diagram bazy danych zawierający tabele i ich połączenia.

Następnie opis bazy, to znaczy tabel, pól w tabelach i ich relacji. W tym samym rozdziele znajdziemy też zapytania do bazy wpisujące rekordy.

Na samym końcu projektu znajduję się kod bazy danych, który jednym uruchomieniem tworzy bazę danych

Schemat


Customers
* (All Columns)
id
customer_name
surname
company_name
company_number
street
street_number
flat_number
post_code
city
phone_number
mail
login
password
Ordered_products
* (All Columns)
id
order_id
product_id
quantity
Orders
* (All Columns)
id
customer_id
date_of_placing_the_order
order_realization_date
whether_the_order_fulfilled
shipping_date
employee_id
invoice_id
Product_category
* (All Columns)
id
category_name
Products
* (All Columns)
id
product_name
category_id
producer_id
available_pieces
net_catalog_price
net_selling_price
net_selling_warehouse
VAT_rate
Employees
* (All Columns)
id
employee_name
employee_surname
mail
login
password
employee_bonus
Producers
* (All Columns)
id
producer_name
Invoices
* (All Columns)
id
invoice_name
date_of_invoice

Opis bazy danych

Każda tabela zawiera pole id które zawiera autoinkrementacje i rekordy w nie zawarte nie mogą się powtarzać.

Tabela klienci:

Przechowuje dane związane z klientami sklepu. Jest też wykorzystywana do systemu logowania. Każdy klient sklepu musi założyć konto, podać dane kontaktowe oraz adres, który będzie służył do wysyłki. Tabela połączona jest z tabela zamówienia za pomocą klucza obcego. Do każdego zamówienia przypisany jest klient , co umożliwia znalezienie zamówienia, które było zrealizowane przez konkretnego klienta.

- customer_name, surname, company_name, company_number: zawiera nazwy klientów, opcjonalnie nazwę firmy i numer nip. Imiona i nazwiska, czy też nazwy firm bywają bardzo długie zważywszy na to można wprowadzić do 50 znaków, pola obowiązkowe.

- street i city: tutaj sytuacja wygląda podobnie możemy wprowadzić nazwy ulic i miejscowości do 50 znaków. Są obowiązkowe

- street_number i flat_number: to cyfry, które nie mają bardzo dużego zakresu. Dodatkowo numer mieszkania może nie istnieć.

- phone_number: numer telefonu zawiera 9 cyfr plus numer kierunkowy i jest obowiązkowy.

- mail: nie może zawierać znaków spoza alfabetu łacińskiego. Dodatkowo pole sprawdza, czy w mailu znajduje się znak jeden znak @. Pole wymagane i unikatowe.

- login: pozwala na wprowadzenie nazwy użytkownika do 50 znaków i jest obowiązkowe. Zawartość nie może się powtarzać.

- password: pole zawiera do 60 znaków i będą się tam znajdowały zaszyfrowane hasła.

Tabela pracownicy:

Podobnie jak tabela klienci, przechowuje najważniejsze dane. W tym przypadku pracowników. Pozwala również zarządzać przyznaną pracownikowi premią, która może być zależna od sprzedaży produktów. Tabela połączona jest z tabela zamówienia za pomocą klucza obcego, dzięki czemu łatwo można sprawdzić, co sprzedał dany pracownik.

- employee_name i employee_surname: pola zawierjące imiona i nazwiska użytkowników, mogą zawierać do 50, znaków są obowiązkowe.

- mail: nie może zawierać znaków spoza alfabetu łacińskiego. Dodatkowo pole sprawdza, czy w mailu znajduje się znak jeden znak @. Pole wymagane i unikatowe.

- login: pozwala na wprowadzenie nazwy użytkownika do 50 znaków i jest obowiązkowe. Zawartość nie może się powtarzać.

- password: pole zawiera do 60 znaków i będą się tam znajdowały zaszyfrowane hasła.

- employee_bonus: pole na podstawie, którego będzie zarządzana premia sprzedawcy

Tabela zamówienia:

W tabeli zamówienia znajdują się najważniejsze informacje związane z zamówieniem. Dzięki tabeli możemy wyszukać, które zamówienie było zrealizowane na jakiego klienta, co może być przydatne w przypadku zwrotu, czy reklamacji. Tabela ta jest głównym członem bazy danych. Połączona jest kluczami obcymi z tabelami klienci, pracownicy, faktury.

- customer_id: pole łączce tabele z tabelą klienci

- date_of_placing_the_order: jest to pole zawierające datę utworzenia zamówienia i jest obowiązkowe, ponieważ przyjęcie zamówienia następuje zaraz po zamówieniu przez klienta poprzez stronę.

- order_realization_date: w polu znajduje się data zrealizowania zamówienia, nie jest to obowiązkowe pole, gdyż zamówienie może zostać anulowane lub nie dojść do skutku przez inne przyczyny zewnętrzne.

- whether_the_order_fulfilled: jest to pole mówiące tylko o stanie zamówienia, czy zostało zrealizowane, czy też nie.

- shipping_date: jest to pole zawierające datę nadania zamówienia do klienta. Może nie zostać wypełnione z tych samych przyczyn co pole order_realization_date.

- employee_id: łączy bieżącą tabele z tabelą pracownicy, co pozwala na przypisanie pracownika do zamówienia.

- invoice_id: pole łączce zamówienia z tabelą faktury, dzięki czemu możemy wyszukać zamówienie przez fakturę, czy też odnaleźć fakturę po id zamówienia

Tabela produkty:

Znajdują się w niej dostępne do sprzedaży produkty ich ceny katalogowe, sprzedażowe oraz dostępne w hurtowni. Na podstawie zależności cen można wyliczyć możliwy utarg ze sprzedaży danego produktu. Tabela połączona jest kluczami obcymi z tabelami kategorie oraz producenci. Wszystkie pola w tabeli są obowiązkowe, ponieważ każdy produkt na magazynie musi mieć swój numer, ilość, wartość i stawkę vat.

- product_name: pole zawierające nazwy produktów. Maksymalna nazwa produktu to 50 znaków.

- category_id: pole łączące tabele z tabelą categorie, ponieważ do produktów przypisane są ich kategorie na stronie, co ułatwia wyszukiwanie.

- producer_id: łączy tabele z tabelą producenci, gdzie znajdują się nazwy producentów danego produktu.

- available_pieces: to pole zawierające ilość produktów dostępnych na magazynie

- net_catalog_price, net_catalog_selling i net_selling_werehouse: to ceny po jakich firma sprzedaje produkty, kupuje z hurtowni oraz katalogowe. Podawane są w wartości dzietnej do dwóch miejsc po przecinku.

- VAT_rate: pole opisujące stawkę VAT na dany produkt

Tabela faktury:

Może zawierać faktury wystawione do zrealizowanych zamówień z tabeli zamówienia. W późniejszym czasie jeśli klient będzie posiadał nazwę faktury, tabela może ułatwić odnajdowanie zamówień w bazie na przykład w razie zwrotu, czy też wysłania na serwis.

- invoice_name: pole zawierające nazwę faktury, nazwa faktury będzie składać się z przedrostka i numeru. Pole nie może się powtarzać. Nie może być puste.

- date_of_invoice: pole zawierające datę wystawienia faktury. Również nie może być puste

Tabela zamówione produkty:

Tabela przejściowa między zamówieniami i produktami. Pozwala na wybieranie produktów i ich ilości do zamówienia. Połączona jest kluczami obcymi z tabelami produkty i zamówienia.

- order_id: pole łączące tabele z tabelą zamówienia

- product_id: pole to łączy z tabelą produkty, dzięki połączeniu wiemy jaka ilość produktu została sprzedana, a jaka dostępna jest na magazynie

- quantuty: pole pokazujące jaka ilość danego produktu została zamówione przez danego klienta

Tabela producenci:

Zawiera nazwy producentów. Może być wykorzystywana do wyszukiwania produktów, które są związane z konkretnym producentem, sortowania po nazwie producenta, czy też podsumowania, który producent najlepiej się sprzedaje.

- producer_name: pole zawierające nazwy producentów produktów. Musi zostać wypełnione

Tabela kategorie:

Zawiera nazwy kategorii produktowych, które dostępne są w sklepie. Może umożliwić sortowanie produktów na stronie po kategorii lub umożliwić wyświetlanie treści w kolejności, która najczęściej jest wybierana(klikana)

category_name: pole zawierające nazwy kategorii produktów. Musi zostać wypełnione

Przykładowe zapytania do bazy

```
USE rtvDatabase;

GO

--1. Zapytnie wypisuję, kto kupił jaki produkt

SELECT  c.customer_name, c.surname, o.customer_id, p.product_name,
p.net_selling_price

FROM   dbo.Customers c

       INNER JOIN dbo.Orders o on c.id = o.id
       INNER JOIN dbo.Products p on p.id = o.id;



-- 2. Zapytanie wypisuje produkt, który nigdy się nie sprzedał

SELECT id, product_name FROM dbo.Products WHERE id not in (SELECT product_id
FROM dbo.Ordered_products)



-- 3. Zapytanie wypisuje zysk sklepu w podanym okresie czasu


SELECT SUM(p.net_selling_price) AS zarobiono

FROM   dbo.Orders c

        INNER JOIN dbo.Products p on p.id = c.id

WHERE order_realization_date BETWEEN '20000101' AND '20191231'



-- 4. Zapytanie podaje bilans sprzedaży dla każdego producenta.

SELECT o.producer_name, sum(p.net_selling_price*c.quantity)

FROM   dbo.Products p

        INNER JOIN dbo.Ordered_products c on p.id = c.product_id
             INNER JOIN dbo.Producers o on p.producer_id = o.id
GROUP BY o.producer_name
```

Kod bazy danych

```
USE master

GO
--Tworzenie bazy
CREATE DATABASE rtvDatabase;
GO

--Tworzenie tabel:

USE rtvDatabase;

CREATE TABLE Product_category(
id INT IDENTITY(1, 1) CONSTRAINT category_id
PRIMARY KEY NOT NULL, category_name NVARCHAR(50) NOT NULL
);

CREATE TABLE Producers(
id INT IDENTITY(1,1) CONSTRAINT producer_id PRIMARY
KEY NOT NULL, producer_name NVARCHAR(50) NOT NULL
);

CREATE TABLE Customers(
id INT IDENTITY(1, 1) CONSTRAINT customer_id PRIMARY
KEY NOT NULL, customer_name NVARCHAR(50) NOT NULL,
surname NVARCHAR(50) NOT NULL, company_name NVARCHAR(50), company_number
TINYINT, street NVARCHAR(50) NOT NULL, street_number
SMALLINT NOT NULL, flat_number SMALLINT, post_code VARCHAR(6) NOT NULL,
city NVARCHAR(50) NOT NULL, phone_number CHAR(12) NOT NULL, mail VARCHAR(50)
NOT
NULL UNIQUE CHECK(((LEN(mail) - LEN(REPLACE(mail, '@', ''))) = 1)),
login NVARCHAR(50) UNIQUE NOT NULL, password CHAR(60) NOT NULL
);

CREATE TABLE Products(
id INT IDENTITY(1, 1) CONSTRAINT product_id PRIMARY KEY
NOT NULL, product_name NVARCHAR(50) NOT NULL,
category_id int CONSTRAINT id_category FOREIGN KEY REFERENCES
Product_category(id) NOT NULL,
producer_id int CONSTRAINT id_producer FOREIGN KEY REFERENCES Producers(id) NOT
NULL, available_pieces SMALLINT NOT NULL,
net_catalog_price DECIMAL(6,2) NOT NULL, net_selling_price DECIMAL(6,2) NOT
NULL, net_selling_warehouse DECIMAL(6,2) NOT NULL, VAT_rate TINYINT NOT NULL
);

CREATE TABLE Employees(
id INT IDENTITY(1, 1) CONSTRAINT employee_id PRIMARY
KEY NOT NULL, employee_name NVARCHAR(50) NOT NULL,
employee_surname NVARCHAR(50) NOT NULL, mail VARCHAR(50) NOT NULL UNIQUE
CHECK(((LEN(mail) - LEN(REPLACE(mail, '@', ''))) = 1)),
login NVARCHAR(50) UNIQUE NOT NULL, password CHAR(60) NOT NULL, employee_bonus
DECIMAL (6,2) DEFAULT 200.00
);

CREATE TABLE Invoices(
id INT IDENTITY(1, 1) CONSTRAINT invoice_id PRIMARY KEY
NOT NULL, invoice_name VARCHAR(12) UNIQUE NOT NULL,
date_of_invoice DATETIME NOT NULL
);
```

```sql
CREATE TABLE Orders(
id INT IDENTITY(1, 1) CONSTRAINT order_id PRIMARY KEY NOT
NULL,
customer_id int CONSTRAINT id_customer FOREIGN KEY REFERENCES Customers(id) NOT
NULL,
date_of_placing_the_order DATETIME NOT NULL, order_realization_date DATETIME,
whether_the_order_fulfilled BIT, shipping_date DATETIME,
employee_id int CONSTRAINT id_employee FOREIGN KEY REFERENCES Employees(id) NOT
NULL,
invoice_id int CONSTRAINT id_invoice FOREIGN KEY REFERENCES Invoices(id) NOT
NULL
);

CREATE TABLE Ordered_products(
id INT IDENTITY(1, 1) CONSTRAINT
orderd_products_id PRIMARY KEY NOT NULL,
order_id int CONSTRAINT id_order FOREIGN KEY REFERENCES Orders(id) NOT NULL,
product_id int CONSTRAINT id_product FOREIGN KEY REFERENCES Products(id) NOT
NULL, quantity INT NOT NULL
);

--Tworzenie rekordów:

INSERT INTO Customers( customer_name, surname, street, street_number,
flat_number, post_code,
city, phone_number, Mail, login, password)
VALUES
('Michal', 'Salek', 'Tetmaiera', 12, null, '31-333', 'Kraków',
'123456789', 'michal@o2.pl', 'michal123', 'qwerty1'),
('Julia', 'Maly', 'Rolna', 34, null, '31-322', 'Kraków', '232145621',
'julia@o2.pl', 'julia123', 'qwerty121'),
('Jan', 'Kowalski', 'Polna', 3, 4, '31-321', 'Kraków', '232124213',
'jan@o2.pl', 'jan123', 'qwerty123'),
('Tomasz', 'Nowak', 'Szkolna', 44, null, '31-123', 'Kraków',
'987654321', 'tomasz@o2.pl', 'tomek123', 'qwerty1234'),
('Pawel', 'Duzy', 'Wolna', 22, null, '31-111', 'Kraków', '321452332',
'pawel@o2.pl', 'pawel123', 'qwerty141');

INSERT INTO Product_category(category_name)
VALUES ('Telewizor'), ('Glosniki'), ('Mikrofalowka'), ('Laptop'),
('Klawiatura');

INSERT INTO Producers(producer_name) VALUES
('Sony'),('Logitech'), ('Lg'), ('Acer'), ('Dell');

INSERT INTO Products( product_name, category_id, producer_id, available_pieces,
net_catalog_price,
net_selling_price, net_selling_warehouse, VAT_rate)
VALUES
('Sony KD-65XF9005', 1, 1, 1, 5500.00, 5300.00, 4900.00, 23),
('Dell Inspiron 3583', 4, 5, 2, 2199.00, 1949.00, 1799.00, 23),
('LG microvawe', 3, 3, 299.00, 2, 299.00, 199.00, 23),
('Logitech K906', 2, 2, 1149, 10, 999.00, 850.00, 23),
('Logitech K120', 5, 2, 89.00, 8, 79.00, 58.00, 23),
('Logitech G502', 2, 2, 249.00, 12, 199.00, 149.00, 23);

INSERT INTO Employees ( employee_name, employee_surname, mail, login, password,
employee_bonus)
VALUES
```

```
('Władysław', 'Łokietek', 'lokietek@gmail.com', 'wlalo123', 'zaq1@WSX',
DEFAULT),
('Władysław', 'Jagieło', 'jakieglo@gmail.com', 'wladzia22', 'zaq1@WSX',
DEFAULT),
('Jan', 'Sobieski', 'sobjan@gmail.com', 'sebek77', 'zaq1@WSX', DEFAULT),
('Jacek', 'Soplica', 'soplicajacek@gmail.com', 'sopjacek99', 'zaq1@WSX',
DEFAULT);

INSERT INTO Invoices( invoice_name, date_of_invoice)
VALUES
('KK2021000001', '20180602'),
('KK2021000002', '20200210'),
('KK2021000003', '20200210'),
('KK2021000004', '20190325'),
('KK2021000005', '20200131');

INSERT INTO Orders ( customer_id, employee_id, invoice_id,
date_of_placing_the_order, order_realization_date, whether_the_order_fulfilled,
shipping_date)
VALUES
(1, 2, 1, '20180530', '20180602', 'true', '20180531'),
(2, 2, 2, '20200202', '20200210', 'false', '20200203'),
(2, 4, 3, '20200202', '20200210', 'false', '20200203'),
(3, 1, 4, '20190322', '20190325', 'true', '20190323'),
(5, 3, 5, '20200129', '20200131', 'true', '20200129');

INSERT INTO Ordered_products (order_id, product_id, quantity)
VALUES
(1, 3, 1),
(2, 3, 2),
(3, 2, 1),
(4, 5, 1),
(5, 6, 2);
```