**HURTOWNIE DANYCH**

Laboratorium 1

Maciej Kopiński 254578

**Zad 1. Modelowanie danych**

Proszę przeanalizować konceptualny model danych „Usługi” (Rys. 1), który jest niekompletny,

ale klasy i relacje między nimi reprezentują rozpatrywany wycinek rzeczywistości. Następnie

proszę wykonać następujące zadania:

1. Zweryfikować model danych w kontekście podanego zbioru reguł i ograniczeń dziedzinowych modyfikując zbiór reguł i ograniczeń (uzupełniając lub poprawiając ich definicję)

2. Przedstawić uzupełnioną i poprawioną wersję modelu danych (kompletny diagram klas

UML)

3. Utworzyć logiczny model danych w postaci skryptu w języku DDL SQL (uwzględniając reguły i ograniczenia dziedzinowe), starając się zachować zgodność ze standardem

języka SQL (pomijając, o ile to możliwe, natywne konstrukcje implementacji języków

SQL)

4. Utworzyć bazę danych w systemie MS SQL 2019, która jest fizycznym modelem danych modelowanego wycinka rzeczywistości

5. Wprowadzić klika rekordów do każdej tabeli sprawdzając poprawność implementacji

(zarówno poprawne dane, jak i niezgodne z obowiązującymi regułami – komentując

i wyjaśniając uzyskane komunikaty z systemu SZBD)

**Zad. 2. Podstawy SQL**

Proszę zapisać zapytania SQL, które dadzą odpowiedź na poniższe pytania. Proszę

zinterpretować wyniki.

1. Ile jest produktów w bazie? Ile kategorii i podkategorii?

2. Wypisz produkty, które nie mają zdefiniowanego koloru.

3. Podaj roczną kwotę transakcji (SalesOrderHeader.TotalDue) w poszczególnych latach.

4. Ilu jest klientów, a ilu sprzedawców w sklepie? Ilu w poszczególnych regionach?

5. Ile było wykonanych transakcji w poszczególnych latach?

6. Podaj produkty, które nie zostały kupione przez żadnego klienta. Zestawienie pogrupuj

według kategorii i podkategorii.

7. Oblicz minimalną i maksymalną kwotę rabatu udzielonego na produkty w poszczególnych podkategoriach.

8. Podaj produkty, których cena jest wyższa od średniej ceny produktów w sklepie.

9. Ile średnio produktów sprzedaje się w poszczególnych miesiącach?

10. Ile średnio czasu klient czeka na dostawę zamówionych produktów? Przygotuj zestawienie w zależności od kodu regionu (SalesTerritory.CountryRegionCode).

**Zad.1**

1.1

Reg/01 – Poprawna.

Reg/02 – W sklepie może robić zakupy dowolna liczba klientów.

Reg/03 – Poprawna.

Reg/04 – Sklep może oferować wiele produktów.

Reg/05 – Produkt może być w wielu sklepach.

Reg/06 – Klient musi nabyć co najmniej jeden produkt.

Reg/07 – Produkt może być nabyty przez wielu klientów.

Reg/08 – Zakup każdego produktu posiada ilość i cenę produktu.

Reg/09 – Oferta produktu posiada ilość oraz cenę produktu.

Ogr/01 – Imię oraz nazwisko klienta nie mogą być puste.

Ogr/02 – Nazwa sklepu nie może być pusta.

Ogr/03 – Liczba produktów musi być większa lub równa 0.

Ogr/04 – Cena produktu musi być większa od 0.

Ogr/05 – Nazwa produktu nie może być pusta.

Ogr/06 – Data zakupów nie może być późniejsza od dnia obecnego.

Ogr/07 – Ilość i cena nabytych produktów muszą być większe od 0.

Ogr/08 – Klient musi posiadać imię oraz nazwisko.

Ogr/09 – Zakupy muszą mieć datę zakupów.

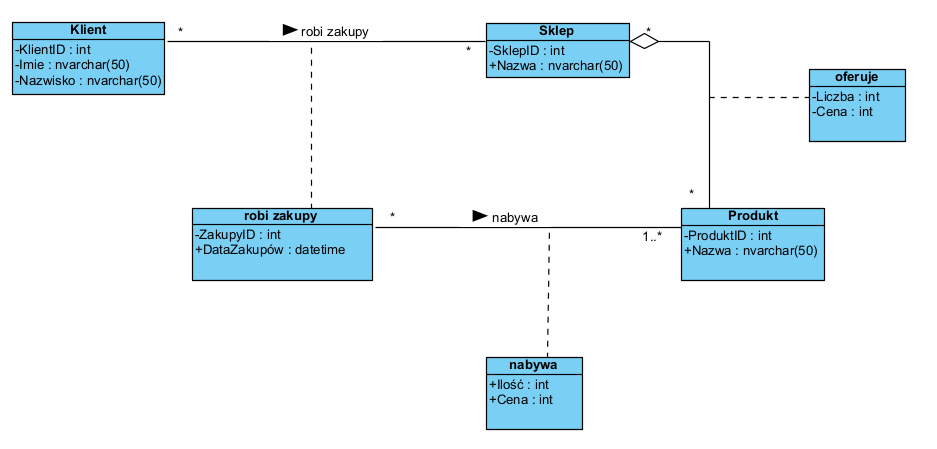
Ogr/10 – Nabycie produktu musi mieć ilość oraz cenę produktu.

Ogr/11 – Produkt musi mieć nazwę.

Ogr/12 – Oferta musi mieć ilość oraz cenę.

Ogr/13 – Sklep musi mieć nazwę.

1.2



1.3

CREATE DATABASE Uslugi;

CREATE TABLE Klient (

KlientID INT PRIMARY KEY IDENTITY(1, 1),

FirstName VARCHAR(50) CHECK(FirstName != '') NOT NULL,

LastName VARCHAR(50) CHECK(LastName != '') NOT NULL,

)

CREATE TABLE Sklep (

SklepID INT PRIMARY KEY IDENTITY(1, 1),

Nazwa NVARCHAR(50) CHECK(Nazwa != '') NOT NULL,

)

CREATE TABLE Produkt (

ProduktID INT PRIMARY KEY IDENTITY(1, 1),

Nazwa NVARCHAR(50) CHECK(Nazwa != '') NOT NULL

)

CREATE TABLE KlientSklep (

KlientSklepID INT PRIMARY KEY IDENTITY(1, 1),

DataZakupow DATETIME CHECK(DataZakupow <= GETDATE()) NOT NULL,

KlientID INT FOREIGN KEY REFERENCES Klient(KlientID) NOT NULL,

SklepID INT FOREIGN KEY REFERENCES Sklep(SklepID) NOT NULL

)

CREATE TABLE SklepProdukt (

SklepID INT FOREIGN KEY REFERENCES Sklep(SklepID),

ProduktID INT FOREIGN KEY REFERENCES Produkt(ProduktID),

Liczba INT CHECK(Liczba >= 0) NOT NULL,

Cena INT CHECK(Cena > 0) NOT NULL

CONSTRAINT PK\_SklepProdukt PRIMARY KEY (SklepID, ProduktID)

)

CREATE TABLE KlientSklepProdukt (

KlientSklepID INT FOREIGN KEY REFERENCES KlientSklep(KlientSklepID),

ProduktID INT FOREIGN KEY REFERENCES Produkt(ProduktID),

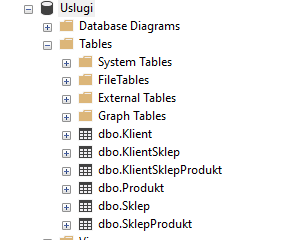
Ilosc INT CHECK(Ilosc > 0) NOT NULL,

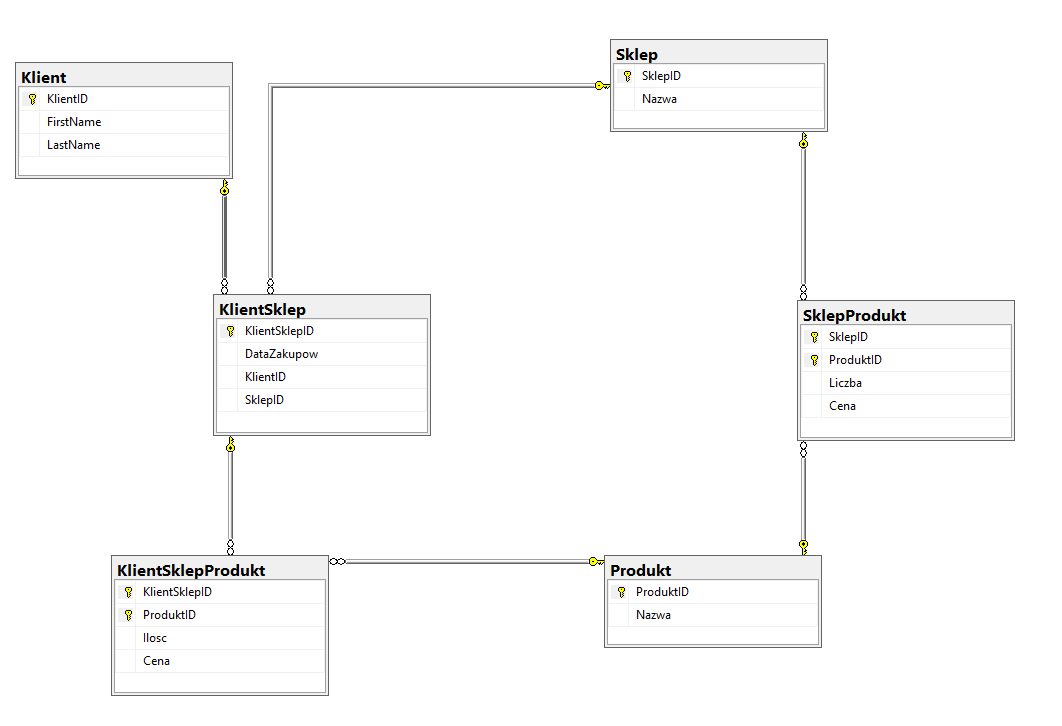
Cena INT CHECK(Cena > 0) NOT NULL,

CONSTRAINT PK\_KlientSklepProdukt PRIMARY KEY (KlientSklepID, ProduktID)

);

1.4



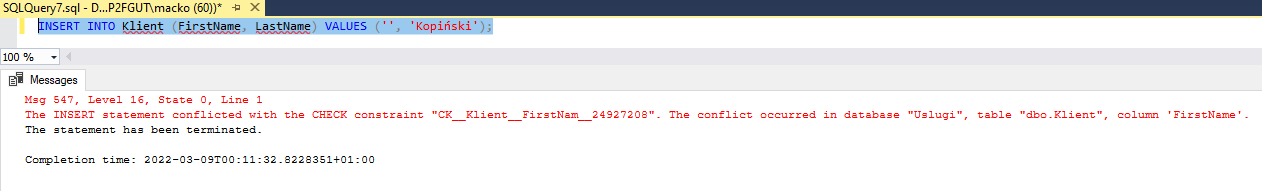


1.5

Sprawdzenie poprawności danych dla tabeli Klient

Obraz zawierający tekst

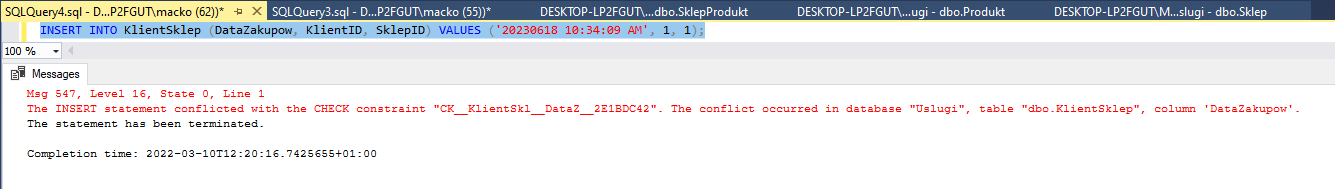
Opis wygenerowany automatycznie

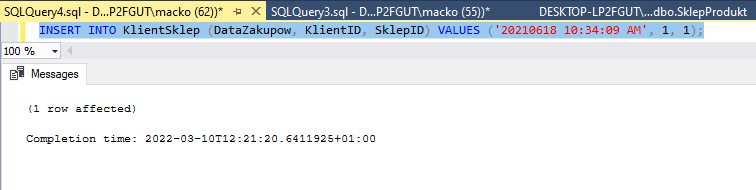


Obraz zawierający tekst

Opis wygenerowany automatycznie

Sprawdzenie poprawności danych dla tabeli KlientSklep (W bazie istnieje klient od id=1 i sklep o id=1)





**Zad.2**

--Pkt1

SELECT COUNT(\*) FROM Production.Product;

SELECT COUNT(\*) FROM Production.ProductCategory;

SELECT COUNT(\*) FROM Production.ProductSubcategory;

-- Pkt2

SELECT \* FROM Production.Product WHERE Product.Color IS NULL;

--Pkt3

SELECT YEAR(Sales.SalesOrderHeader.OrderDate), SUM(Sales.SalesOrderHeader.TotalDue) FROM Sales.SalesOrderHeader GROUP BY YEAR(Sales.SalesOrderHeader.OrderDate);

--Pkt4

SELECT COUNT(\*) FROM Sales.SalesPerson;

SELECT COUNT(\*) FROM Sales.Customer;

SELECT Sales.SalesTerritory.Name, COUNT(\*)

FROM Sales.SalesPerson

FULL OUTER JOIN Sales.SalesTerritory ON Sales.SalesPerson.TerritoryID = Sales.SalesTerritory.TerritoryID

GROUP BY Sales.SalesTerritory.Name;

SELECT Sales.SalesTerritory.Name, COUNT(\*)

FROM Sales.Customer

FULL OUTER JOIN Sales.SalesTerritory ON Sales.Customer.TerritoryID = Sales.SalesTerritory.TerritoryID

GROUP BY Sales.SalesTerritory.Name;

--Pkt5

SELECT YEAR(Sales.SalesOrderHeader.OrderDate), COUNT(\*) FROM Sales.SalesOrderHeader GROUP BY YEAR(Sales.SalesOrderHeader.OrderDate);

--Pkt6

SELECT Production.ProductCategory.Name, Production.ProductSubcategory.Name, Production.Product.\*

FROM Production.Product

LEFT JOIN Sales.SalesOrderDetail ON Sales.SalesOrderDetail.ProductID = Production.Product.ProductID

LEFT JOIN Production.ProductSubcategory ON Production.Product.ProductSubcategoryID = Production.ProductSubcategory.ProductSubcategoryID

LEFT JOIN Production.ProductCategory ON Production.ProductSubcategory.ProductCategoryID = Production.ProductCategory.ProductCategoryID

WHERE Sales.SalesOrderDetail.SalesOrderID IS NULL

ORDER BY Production.ProductCategory.Name, Production.ProductSubcategory.Name;

--Pkt7

SELECT Production.ProductSubcategory.Name, MIN(Sales.SalesOrderDetail.UnitPriceDiscount), MAX(Sales.SalesOrderDetail.UnitPriceDiscount)

FROM Production.Product

JOIN Sales.SalesOrderDetail ON Sales.SalesOrderDetail.ProductID = Production.Product.ProductID

JOIN Production.ProductSubcategory ON Production.Product.ProductSubcategoryID = Production.ProductSubcategory.ProductSubcategoryID

GROUP BY Production.ProductSubcategory.Name;

--Pkt8

SELECT Production.Product.\*

FROM Production.Product

WHERE Production.Product.ListPrice > (SELECT AVG(Production.Product.ListPrice) FROM Production.Product);

--Pkt9

SELECT M, AVG(Q)

FROM (SELECT YEAR(H.OrderDate) Y, MONTH(H.OrderDate) M, SUM(D.OrderQty) Q

FROM Sales.SalesOrderDetail D

JOIN Sales.SalesOrderHeader H ON H.SalesOrderID = D.SalesOrderID

GROUP BY YEAR(H.OrderDate), MONTH(H.OrderDate)) S

GROUP BY M

ORDER BY 1;

--Pkt10

SELECT Sales.SalesTerritory.CountryRegionCode, AVG(CAST(DATEDIFF(day, Sales.SalesOrderHeader.OrderDate, Sales.SalesOrderHeader.ShipDate) AS float))

FROM Sales.SalesOrderHeader

JOIN Sales.SalesTerritory ON Sales.SalesTerritory.TerritoryID = Sales.SalesOrderHeader.TerritoryID

GROUP BY Sales.SalesTerritory.CountryRegionCode;

**Zadanie na zajęciach – Przygotować zestawienie stażu pracowników, kategorii oraz liczby produktów sprzedanych z tej kategorii.**

SELECT DATEDIFF(year, HumanResources.Employee.HireDate, GETDATE()) AS 'Staż', Production.ProductCategory.Name AS 'Kategoria', SUM(Sales.SalesOrderDetail.OrderQty) AS 'Liczba sprzedanych produktów'

FROM HumanResources.Employee

JOIN Sales.SalesPerson ON Sales.SalesPerson.BusinessEntityID = HumanResources.Employee.BusinessEntityID

JOIN Sales.SalesOrderHeader ON Sales.SalesOrderHeader.SalesPersonID = Sales.SalesPerson.BusinessEntityID

JOIN Sales.SalesOrderDetail ON Sales.SalesOrderDetail.SalesOrderID = Sales.SalesOrderHeader.SalesOrderID

JOIN Production.Product ON Production.Product.ProductID = Sales.SalesOrderDetail.ProductID

JOIN Production.ProductSubcategory ON Production.ProductSubcategory.ProductSubcategoryID = Production.Product.ProductSubcategoryID

JOIN Production.ProductCategory ON Production.ProductCategory.ProductCategoryID = Production.ProductSubcategory.ProductCategoryID

GROUP BY DATEDIFF(year, HumanResources.Employee.HireDate, GETDATE()), Production.ProductCategory.Name

ORDER BY 1, 3;

Wynik działania zapytania:

Obraz zawierający stół

Opis wygenerowany automatycznie

**Wnioski:**

Można zauważyć, że wraz ze wzrostem stażu rośnie ilość sprzedawanych produktów. Oznacza to, że nowi pracownicy gorzej radzą sobie od tych bardziej doświadczonych

**Wnioski:**

Ad.1

Model danych zaprezentowany w pierwszym zadaniu, nie licząc pól w klasach miał moim zdaniem jedynie jedną wadę – brak klasy asocjacyjnej „oferuje”, ponieważ relacja pomiędzy sklepem a produktem była relacją wiele do wielu. Po jej dodaniu wymagała ona także uwzględnienia pól ‘liczba’ oraz ‘cena’. Klasa ta reprezentuje ofertę danego sklepu – ile danego produktu oraz w jakiej cenie jest w danym sklepie.

Aby móc przenieść ten model na fizyczną implementację wymagane było dodanie dodatkowych pól, takich jak ‘id’ czy ‘nazwa’, lub ‘imię’ oraz ‘nazwisko’ w przypadku klienta.

Ostatnią rzeczą było usunięcie pola ‘czas’ z klasy asocjacyjnej „robi zakupy” i użycie typu danych datetime.

Reguły biznesowe wymagały także poprawy reguły nr 2 – „W sklepie może robić zakupy dowolny klient” – która sama w sobie nic za bardzo nie określa oraz uzupełnienia o liczności w asocjacjach pomiędzy klasami i pola w klasie asocjacyjnej „oferuje”.

Ad.2

Jeśli chodzi o zapytania z zadania 2, można powiedzieć, że:

* W bazie są 504 produkty, 4 kategorie i 37 podkategorii
* Około połowa produktów nie posiada określonego koloru (248)
* Największa kwota transakcji wystąpiła w 2013 roku, a najmniejsza w 2011 (należy wziąć pod uwagę, że dane z 2014 roku dotyczą jedynie połowy roku)
* W bazie istnieje 17 pracowników, 19820 klientów, a 3 pracowników nie ma przypisanego regionu
* Najwięcej transakcji było wykonane w 2013 roku, a najmniej w 2011 (widać to w podpunkcie 2)
* Klienci nie kupowali 238 różnych produktów
* Największe zniżki były udzielane na rowery
* Wszystkie produkty, których cena jest większa niż średnia cena produktów nie były przeceniane oraz istnieje o wiele więcej produktów tańszych niż średnia cena produktów od produktów droższych niż ta cena
* Najwięcej produktów sprzedaje się w marcu, a najmniej w lutym
* Średnio najdłużej na zamówienie trzeba czekać w Niemczech