

4. sprawozdanie z laboratorium Hurtownie Danych

Mikołaj Kubś, 272662

11 kwietnia 2025

1. DDL (Data Definition Language) - tworzenie struktur danych i schema z użyciem poleceń CREATE, ALTER, DROP
2. DML (Data Manipulation Language) - manipulacja danymi w tabelach z użyciem poleceń INSERT, UPDATE, DELETE
3. DCL (Data Control Language) - zarządzanie uprawnieniami, za pomocą poleceń GRANT, DENY, REVOKE
4. DQL (Data Query Language) - pozyskiwanie danych z bazy, za pomocą polecenia SELECT

1 Zadanie 1 - przygotowanie schematu

```
1 CREATE SCHEMA Kubs
```

Utworzenie dedykowanego schematu pozwala na logiczne odseparowanie obiektów stworzonych na potrzeby laboratorium od pozostałych struktur bazy danych.

2 Zadanie 2 - Tworzenie tabel wymiarów i tabeli faktów

```
1  
2 CREATE TABLE Kubs.DIM_CUSTOMER (  
3     CustomerID INT NOT NULL,  
4     FirstName NVARCHAR(50) NULL,  
5     LastName NVARCHAR(50) NULL,
```

```

6      Title NVARCHAR(8) NULL,
7      City NVARCHAR(30) NULL,
8      TerritoryName NVARCHAR(50) NULL,
9      CountryRegionCode NVARCHAR(3) NULL,
10     [Group] NVARCHAR(50) NULL
11 );
12
13 CREATE TABLE Kubs.DIM_PRODUCT (
14     ProductID INT NOT NULL,
15     Name NVARCHAR(50) NOT NULL,
16     ListPrice MONEY NULL,
17     Color NVARCHAR(15) NULL,
18     SubCategoryName NVARCHAR(50) NULL,
19     CategoryName NVARCHAR(50) NULL,
20     Weight DECIMAL(8, 2) NULL,
21     Size NVARCHAR(5) NULL,
22     IsPurchased BIT NULL DEFAULT 0
23 );
24
25 CREATE TABLE Kubs.DIM SALESPERSON (
26     SalesPersonID INT NOT NULL,
27     FirstName NVARCHAR(50) NULL,
28     LastName NVARCHAR(50) NULL,
29     Title NVARCHAR(8) NULL,
30     Gender NCHAR(1) NULL,
31     CountryRegionCode NVARCHAR(3) NULL,
32     [Group] NVARCHAR(50) NULL
33 );
34
35 CREATE TABLE Kubs.FACT_SALES (
36     ProductID INT NOT NULL,
37     CustomerID INT NOT NULL,
38     SalesPersonID INT NULL,
39     OrderDate INT NOT NULL,
40     ShipDate INT NULL,
41     OrderQty SMALLINT NOT NULL,
42     UnitPrice MONEY NOT NULL,
43     UnitPriceDiscount DECIMAL(8, 4) NOT NULL DEFAULT 0,
44     LineTotal DECIMAL(19, 4) NOT NULL
45 );

```

Zgodnie z poleceniem, kolumny OrderDate oraz ShipDate będą przechowywać dane typu całkowitego. W tabeli faktów sprzedawca nie jest wymagany. W tabeli wymiaru dla produktu kategoria i podkategoria również mogą

być NULL. Tabele DIM_ pełnią oczywiście role wymiarów, a FACT_SALES jest tabelą faktów zawierającą miary dotyczące transakcji sprzedaży.

Wartości NULL w niektórych kolumnach wynikają z opcjonalności tych danych w oryginalnym źródle danych lub mogą być nieobecne (również z powodu braku połączenia z inną tabelą, np. produkt bez podkategorii nie może mieć kategorii).

3 Zadanie 3 - wypełnianie danych

```
1  INSERT INTO Kubs.DIM_CUSTOMER (
2      CustomerID,
3      FirstName,
4      LastName,
5      Title,
6      City,
7      TerritoryName,
8      CountryRegionCode,
9      [Group]
10 )
11 SELECT
12     c.CustomerID,
13     MIN(p.FirstName) AS FirstName,
14     MIN(p.LastName) AS LastName,
15     MIN(p.Title) AS Title,
16     MIN(a.City) AS City,
17     MIN(st.Name) AS TerritoryName,
18     MIN(st.CountryRegionCode) AS CountryRegionCode,
19     MIN(st.[Group]) AS [Group]
20 FROM Sales.Customer AS c
21 LEFT JOIN Person.Person AS p
22     ON c.PersonID = p.BusinessEntityID
23 LEFT JOIN Sales.SalesTerritory AS st
24     ON c.TerritoryID = st.TerritoryID
25 LEFT JOIN Person.BusinessEntityAddress bea
26     ON p.BusinessEntityID = bea.BusinessEntityID
27 LEFT JOIN Person.Address AS a ON bea.AddressID = a.AddressID
28 WHERE c.PersonID IS NOT NULL
29 GROUP BY c.CustomerID;
30
31 INSERT INTO Kubs.DIM_PRODUCT (
32     ProductID,
33     Name,
```

```

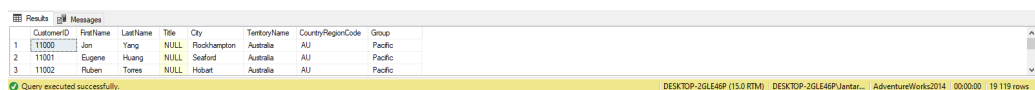
34     ListPrice,
35     Color,
36     SubCategoryName,
37     CategoryName,
38     Weight,
39     Size,
40     IsPurchased
41 )
42 SELECT DISTINCT
43     p.ProductID,
44     p.Name,
45     p.ListPrice,
46     p.Color,
47     psc.Name AS SubCategoryName,
48     pc.Name AS CategoryName,
49     p.Weight,
50     p.Size,
51     1 AS IsPurchased
52 FROM Production.Product AS p
53 INNER JOIN Sales.SalesOrderDetail AS sod
54     ON p.ProductID = sod.ProductID
55 LEFT JOIN Production.ProductSubcategory AS psc
56     ON p.ProductSubcategoryID = psc.ProductSubcategoryID
57 LEFT JOIN Production.ProductCategory AS pc
58     ON psc.ProductCategoryID = pc.ProductCategoryID;
59
60 INSERT INTO Kubs.DIM_SALESPERSON (
61     SalesPersonID,
62     FirstName,
63     LastName,
64     Title,
65     Gender,
66     CountryRegionCode,
67     [Group]
68 )
69 SELECT
70     sp.BusinessEntityID AS SalesPersonID,
71     p.FirstName,
72     p.LastName,
73     p.Title,
74     e.Gender,
75     st.CountryRegionCode,
76     st.[Group]

```

```

77 FROM Sales.SalesPerson AS sp
78 INNER JOIN Person.Person AS p
79     ON sp.BusinessEntityID = p.BusinessEntityID
80 INNER JOIN HumanResources.Employee AS e
81     ON sp.BusinessEntityID = e.BusinessEntityID
82 LEFT JOIN Sales.SalesTerritory AS st
83     ON sp.TerritoryID = st.TerritoryID;
84
85 INSERT INTO Kubs.FACT_SALES (
86     ProductID,
87     CustomerID,
88     SalesPersonID,
89     OrderDate,
90     ShipDate,
91     OrderQty,
92     UnitPrice,
93     UnitPriceDiscount,
94     LineTotal
95 )
96 SELECT
97     sod.ProductID,
98     soh.CustomerID,
99     soh.SalesPersonID,
100     DATEPART(YEAR, soh.OrderDate) * 10000 +
101     DATEPART(MONTH, soh.OrderDate) * 100 +
102     DATEPART(DAY, soh.OrderDate) AS OrderDate,
103     DATEPART(YEAR, soh.ShipDate) * 10000 +
104     DATEPART(MONTH, soh.ShipDate) * 100 +
105     DATEPART(DAY, soh.ShipDate) AS ShipDate,
106     sod.OrderQty,
107     sod.UnitPrice,
108     sod.UnitPriceDiscount,
109     sod.LineTotal
110 FROM Sales.SalesOrderDetail AS sod
111 INNER JOIN Sales.SalesOrderHeader AS soh
112     ON sod.SalesOrderID = soh.SalesOrderID;

```



	CustomerID	FirstName	LastName	Title	City	TerritoryName	County/RegionCode	Group
1	11000	Jon	Yang	NULL	Rockhampton	Australia	AU	Pacific
2	11001	Eugene	Huang	NULL	Seaford	Australia	AU	Pacific
3	11002	Huben	Torres	NULL	Indebat	Australia	AU	Pacific

Query executed successfully. DESKTOP-2GLE48P (15.0 RTM) DESKTOP-2GLE48P\jantar... AdventureWorks2014 00:00:00 19119 rows

Rysunek 1: Pierwsze 3 wiersze DIM_CUSTOMER i liczba wierszy - 19119

ProductID	Name	ListPrice	Color	SubCategoryName	CategoryName	Weight	Size	IsPurchased
984	Mountain-500 Silver, 40	564.99	Silver	Mountain Bikes	Bikes	27.35	40	1
722	LL Road Frame - Black, 58	337.22	Black	Road Frames	Components	2.46	58	1
864	Classic Vest, S	63.50	Blue	Vests	Clothing	NULL	S	1

Rysunek 2: Pierwsze 3 wiersze DIM_PRODUCT i liczba wierszy - 266

SalesPersonID	First Name	Last Name	Title	Gender	Country/Region Code	Group
274	Stephen	Jiang	NULL	M	NULL	NULL
275	Michael	Blythe	NULL	M	US	North America
276	Linda	Mitchell	NULL	F	US	North America

Rysunek 3: Pierwsze 3 wiersze DIM_SALESPERSON i liczba wierszy - 17

ProductID	CustomerID	SalesPersonID	OrderDate	ShipDate	OrderQty	UnitPrice	UnitPriceDiscount	LineTotal
870	29485	276	20140129	20140205	5	2.994	0.0000	14.9700
864	29485	276	20140129	20140205	4	35.10	0.0000	152.4000
870	29485	276	20140129	20140205	3	728.91	0.0000	2186.7300

Rysunek 4: Pierwsze 3 wiersze FACT_SALES i liczba wierszy - 121317

Zgodnie z poleceniem wypełniono tabele danymi z bazy danych AdventureWorks2014.

Ponieważ analizy będą dotyczyć sprzedaży produktów, pominięto wszystkie produkty, które nigdy nie zostały kupione. Okazało się, że wszystkie produkty, które nie miały kategorii lub podkategorii, nigdy nie były kupione.

Potem okazało się, że jest parę wierszy dla klientów z tym samym CustomerID. W takim wypadku wstawiono tylko jeden wiersz.

Liczby wierszy:

DIM_CUSTOMER - 19119

DIM_PRODUCT - 266

DIM_SALESPERSON - 17

FACT_SALES - 121317

4 Zadanie 4 - Więzy integralności

4.1 Definiowanie kluczy głównych i obcych

Poniższy kod SQL dodaje klucze główne do tabel wymiarów oraz klucze obce do tabeli faktów, łącząc ją z wymiarami.

```

1 ALTER TABLE Kubs.DIM_CUSTOMER
2 ADD CONSTRAINT PK_DIM_CUSTOMER PRIMARY KEY (CustomerID);
3
4 ALTER TABLE Kubs.DIM_PRODUCT
5 ADD CONSTRAINT PK_DIM_PRODUCT PRIMARY KEY (ProductID);
6
7 ALTER TABLE Kubs.DIM SALESPERSON
8 ADD CONSTRAINT PK_DIM SALESPERSON PRIMARY KEY (SalesPersonID);
9
10 ALTER TABLE Kubs.FACT_SALES
11 ADD CONSTRAINT FK_FACT_SALES_DIM_CUSTOMER
12 FOREIGN KEY (CustomerID) REFERENCES Kubs.DIM_CUSTOMER(CustomerID);
13
14 ALTER TABLE Kubs.FACT_SALES
15 ADD CONSTRAINT FK_FACT_SALES_DIM_PRODUCT
16 FOREIGN KEY (ProductID) REFERENCES Kubs.DIM_PRODUCT(ProductID);
17
18 ALTER TABLE Kubs.FACT_SALES
19 ADD CONSTRAINT FK_FACT_SALES_DIM SALESPERSON
20 FOREIGN KEY (SalesPersonID)
21 REFERENCES Kubs.DIM SALESPERSON(SalesPersonID);

```

4.2 Testowanie więzów integralności

Poniższe instrukcje INSERT INTO mają na celu sprawdzenie działania zdefiniowanych kluczy głównych i obcych. Oczekujemy, że próby wstawienia niepoprawnych danych zakończą się błędami przechwyconymi w blokach CATCH.

```

1 PRINT 'Test 1: Próba naruszenia PK w DIM_CUSTOMER';
2 BEGIN TRY
3     INSERT INTO Kubs.DIM_CUSTOMER (CustomerID, FirstName, LastName)
4     VALUES (11000, 'Test', 'DuplicatePK');
5
6     PRINT 'BŁĄD - duplikat PK';
7 END TRY
8 BEGIN CATCH
9     PRINT 'SUKCES';
10    PRINT ERROR_MESSAGE();
11 END CATCH
12 GO
13
14 PRINT 'Test 2: Próba naruszenia FK (nieistniejący ProductID) w FACT_SALES';
15 BEGIN TRY

```

```

16     INSERT INTO Kubs.FACT_SALES (
17         ProductID, CustomerID, SalesPersonID, OrderDate, ShipDate,
18         OrderQty, UnitPrice, UnitPriceDiscount, LineTotal
19     ) VALUES (
20         -999,
21         11000,
22         NULL,
23         20250101, 20250102, 1, 10.0, 0, 10.0
24     );
25
26     PRINT 'BŁĄD: Nieistniejące ProductID';
27 END TRY
28 BEGIN CATCH
29     PRINT 'SUKCES';
30     PRINT ERROR_MESSAGE();
31 END CATCH
32 GO
33
34 PRINT 'Test 3: Próba naruszenia FK (nieistniejący CustomerID) w FACT_SALES';
35 BEGIN TRY
36     INSERT INTO Kubs.FACT_SALES (
37         ProductID, CustomerID, SalesPersonID, OrderDate, ShipDate,
38         OrderQty, UnitPrice, UnitPriceDiscount, LineTotal
39     ) VALUES (
40         776,
41         -999,
42         NULL,
43         20240103, 20240104, 2, 20.0, 0, 40.0
44     );
45
46     PRINT 'BŁĄD: nieistniejący CustomerID';
47 END TRY
48 BEGIN CATCH
49     PRINT 'SUKCES';
50     PRINT ERROR_MESSAGE();
51 END CATCH
52 GO
53
54 PRINT 'Koniec testów'

```



```

Messages
Test 1: Próba naruszenia PK w DIM_CUSTOMER
(0 rows affected)
SUCCESS
Violation of PRIMARY KEY constraint 'PK_DIM_CUSTOMER'. Cannot insert duplicate key in object 'Kuba.DIM_CUSTOMER'. The duplicate key value is (11000).
Test 2: Próba naruszenia FK (nieistniejący ProductID) w FACT_SALES
(0 rows affected)
SUCCESS
The INSERT statement conflicted with the FOREIGN KEY constraint "FK_FACT_SALES_DIM_PRODUCT". The conflict occurred in database "AdventureWorks2014", table "Kuba.DIM_PRODUCT", column "ProductID".
Test 3: Próba naruszenia FK (nieistniejący CustomerID) w FACT_SALES
(0 rows affected)
SUCCESS
The INSERT statement conflicted with the FOREIGN KEY constraint "FK_FACT_SALES_DIM_CUSTOMER". The conflict occurred in database "AdventureWorks2014", table "Kuba.DIM_CUSTOMER", column "CustomerID".
Koniec testów

```

Rysunek 5: Wykonanie testów więzów integralności

Testowe instrukcje zakończyły się przechwyceniem błędów związanych z naruszeniem więzów integralności.

5 Zadanie 5 - tworzenie kostki

W środowisku Visual Studio utworzono nowy projekt typu "Analysis Services Multidimensional and Data Mining Project" o nazwie "FirstCube".

Następnie skonfigurowano źródło danych (Data Source), wskazując na bazę AdventureWorks i wykorzystując uwierzytelnianie systemu Windows. Informacje o personifikacji (Impersonation Information) ustawiono na użycie konta usługi (Use the service account), co wymagało nadania odpowiednich uprawnień (db_datareader) temu kontu w bazie SQL Server. Pozwoliło to w zadaniu 6. na poprawne wykonanie Process...

```

1 ALTER ROLE db_datareader ADD MEMBER [NT Service\MSSQLServerOLAPService];
2

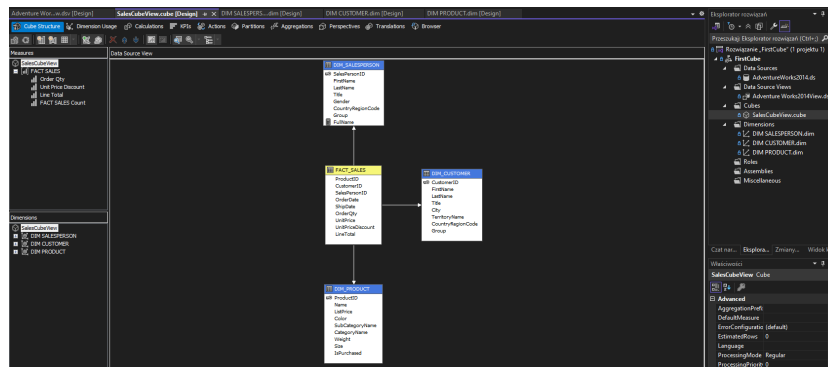
```

Utworzono widok źródła danych (Data Source View), dodając tabele wymiarów oraz tabelę faktów.

Za pomocą kreatora utworzono nową kostkę o nazwie SalesCubeView, bazując na istniejących tabelach z widoku źródła danych:

- Tabela FACT_SALES została wskazana jako tabela faktów (grupa miar).
- Wybrano miary: OrderQty, UnitPriceDiscount oraz LineTotal. Pozostałe kolumny tabeli faktów (ProductID, CustomerID, SalesPersonID, OrderDate, ShipDate) nie są miarami, ponieważ pełnią rolę kluczy obcych do wymiarów lub przechowują informacje opisowe (daty), których agregacja numeryczna (np. suma dat) nie ma sensu biznesowego. Miary reprezentują wartości ilościowe lub pieniężne, które można agregować (sumować, liczyć, uśredniać itp.).
- Tabele DIM_CUSTOMER, DIM_PRODUCT, DIM SALESPERSON zostały wybrane jako wymiary kostki.

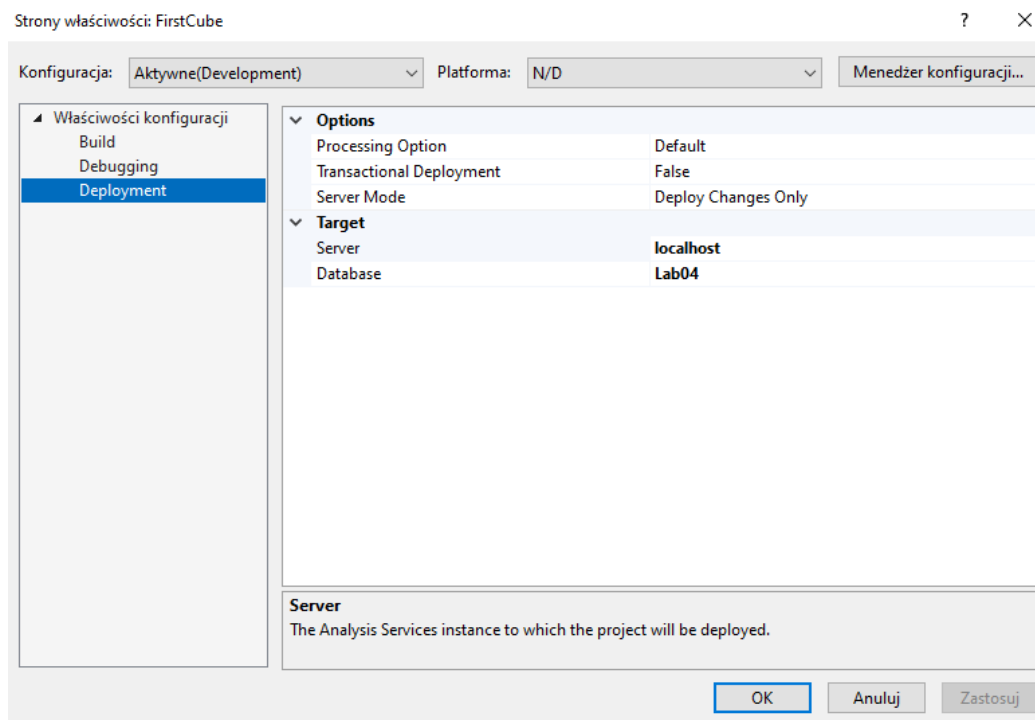
Po utworzeniu kostki dokonano edycji wymiarów, dodając do nich odpowiednie atrybuty.



Rysunek 6: Struktura kostki w edytorze Visual Studio

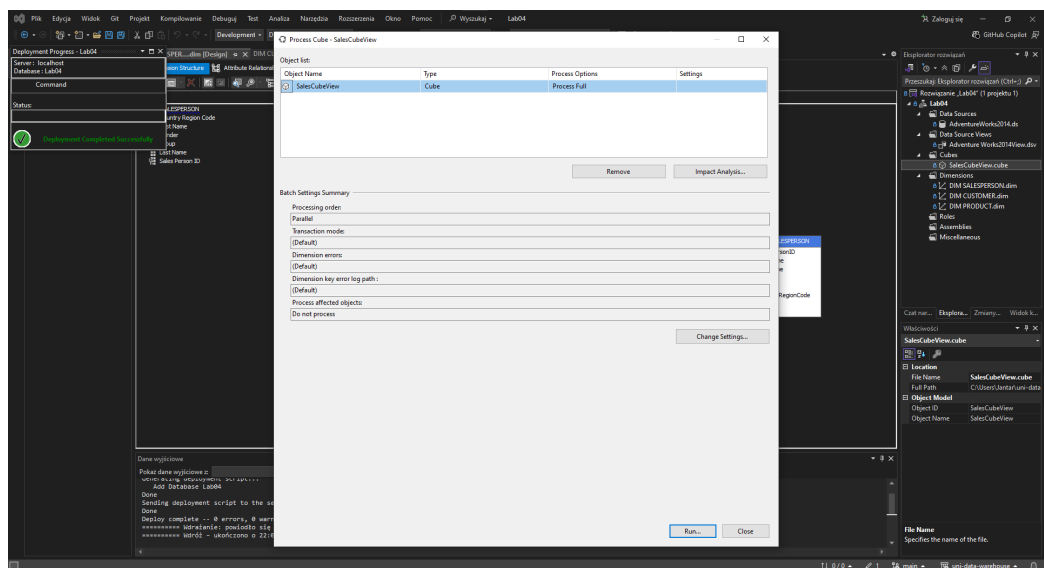
6 Zadanie 6 - Uruchomienie kostki

Jako serwer docelowy (Target Server) ustawiono localhost i podano nazwę docelowej bazy danych SSAS Lab04.



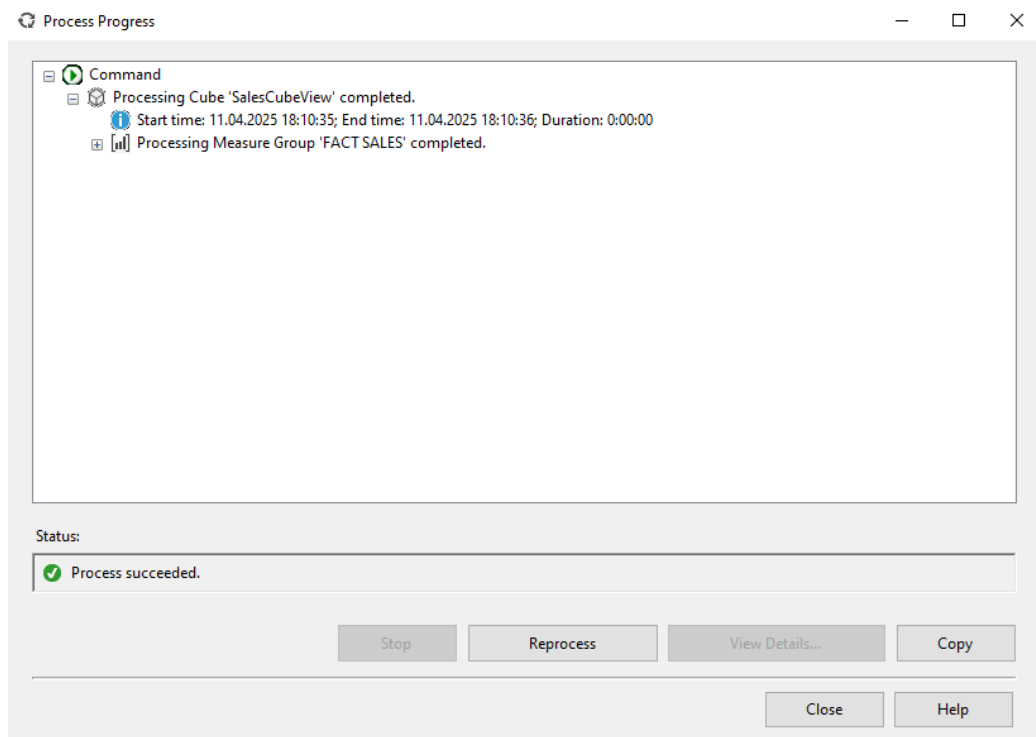
Rysunek 7: Ustawienia konfiguracji wdrożenia

Następnie wdrożono projekt za pomocą opcji Build -> Deploy Solution. Wdrożenie zakończyło się sukcesem.



Rysunek 8: Udany deployment

Kolejnym krokiem było przetworzenie kostki. Kliknięto prawym przyciskiem na kostkę w **Solution Explorer** i wybrano opcję **Process...** (jedną z 4). Proces przetwarzania również zakończył się powodzeniem (choć dopiero po nadaniu poprawnych uprawnień).



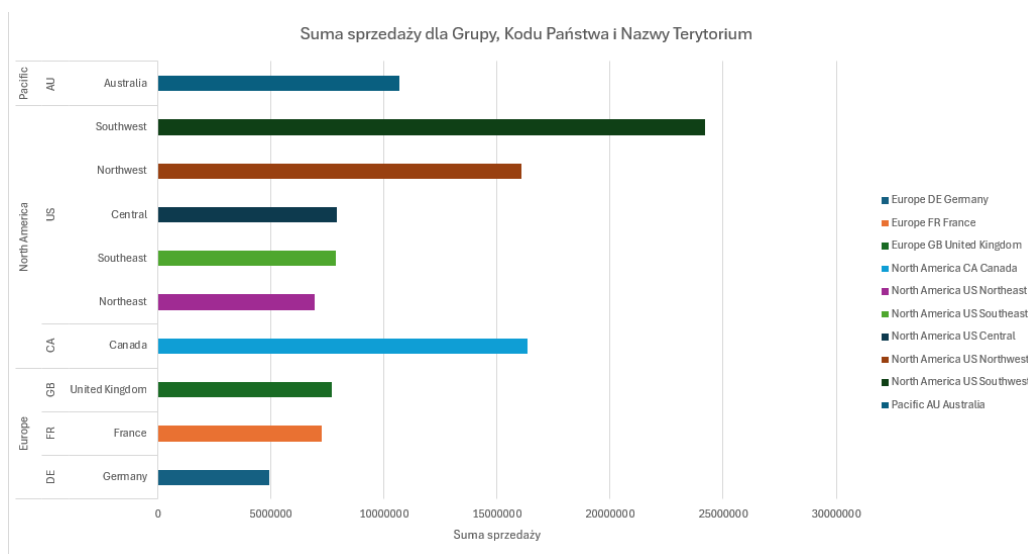
Rysunek 9: Potwierdzenie pomyślnego przetworzenia kostki

7 Zadanie 7 - Proste raporty

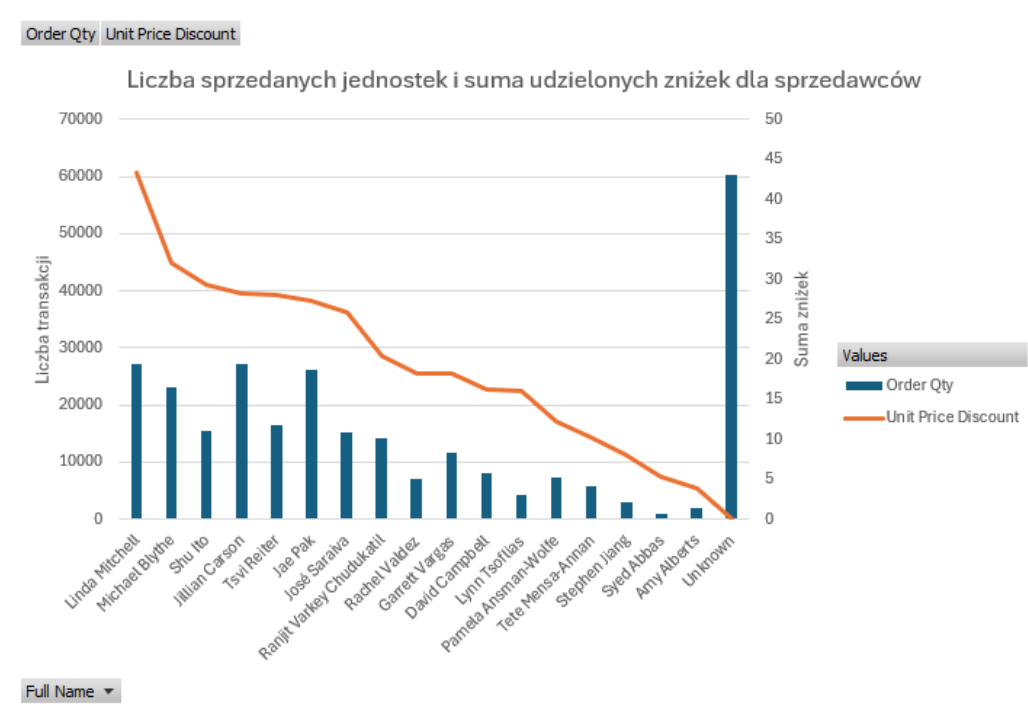
Do analizy danych zawartych w kostce wykorzystano program MS Excel. Połączono się z przetworzoną kostką Analysis Services, wybierając opcję w GUI Visual Studio wewnątrz SalesCubeView -> Browser -> Analizie in Excel.

Przeprowadzono przykładowe analizy, tworząc tabele przestawne i wykresy na ich podstawie:

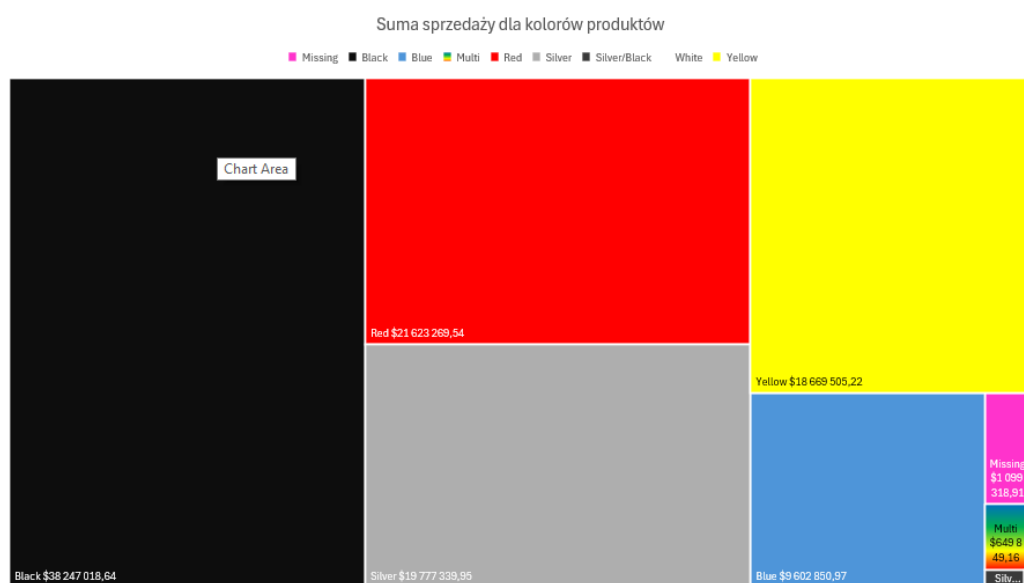
- Analiza sumy sprzedaży (**Line Total**) według hierarchii geograficznej klientów (Grupa -> Kod Państwa -> Nazwa Terytorium).
- Analiza liczby sprzedanych sztuk (**Order Qty**) i sumę udzielonych zniżek (**Unit Price Discount**) według sprzedawców.
- Analiza sumy sprzedaży (**Line Total**) w zależności od koloru produktu (**Color**).



Rysunek 10: Analiza sumy sprzedaży (Line Total) według hierarchii geograficznej klientów (Grupa -> Kod Państwa -> Nazwa Terytorium).



Rysunek 11: Analiza liczby sprzedanych sztuk (Order Qty) i sumę udzielonych zniżek (Unit Price Discount) według sprzedawców.



Rysunek 12: Analiza sumy sprzedaży (Line Total) w zależności od koloru produktu (Color)