5. sprawozdanie z laboratorium Hurtownie Danych

Mikołaj Kubś, 272662

28 kwietnia 2025

1 Zadanie 1 - Przygotowanie powtarzalności procesu ETL

W tym zadaniu celem było przygotowanie skryptu SQL, który umożliwi wielokrotne uruchamianie procesu ETL poprzez usunięcie istniejących tabel wymiarów, faktów oraz tabel pomocniczych, wraz z powiązanymi więzami integralności (foreign keys). Wykorzystano instrukcje ‘DROP TABLE IF EXISTS‘ oraz warunkowe usuwanie ograniczeń za pomocą ‘IF EXISTS‘ i zapytania do ‘INFORMATION_SCHEMA.TABLE_CONSTRAINTS‘.

```
IF EXISTS (
    SELECT
        *
    FROM
        INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    WHERE
        CONSTRAINT_SCHEMA = 'Kubs'
        AND CONSTRAINT_NAME = 'FK_FACT_SALES_DIM_CUSTOMER'
        AND TABLE_NAME = 'FACT_SALES'
)
ALTER TABLE
    Kubs.FACT_SALES DROP CONSTRAINT FK_FACT_SALES_DIM_CUSTOMER;

IF EXISTS (
    SELECT
        *
    FROM
        INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    WHERE
```

```
        CONSTRAINT_SCHEMA = 'Kubs'
        AND CONSTRAINT_NAME = 'FK_FACT_SALES_DIM_PRODUCT'
        AND TABLE_NAME = 'FACT_SALES'
)
ALTER TABLE
    Kubs.FACT_SALES DROP CONSTRAINT FK_FACT_SALES_DIM_PRODUCT;

IF EXISTS (
    SELECT
        *
    FROM
        INFORMATION_SCHEMA.TABLE_CONSTRAINTS
    WHERE
        CONSTRAINT_SCHEMA = 'Kubs'
        AND CONSTRAINT_NAME = 'FK_FACT_SALES_DIM_SALESPERSON'
        AND TABLE_NAME = 'FACT_SALES'
)
ALTER TABLE
    Kubs.FACT_SALES DROP CONSTRAINT FK_FACT_SALES_DIM_SALESPERSON;

DROP TABLE IF EXISTS Kubs.FACT_SALES;

DROP TABLE IF EXISTS Kubs.DIM_CUSTOMER;

DROP TABLE IF EXISTS Kubs.DIM_PRODUCT;

DROP TABLE IF EXISTS Kubs.DIM_SALESPERSON;

DROP TABLE IF EXISTS Kubs.DIM_TIME;

DROP TABLE IF EXISTS Kubs.Helper_Months;

DROP TABLE IF EXISTS Kubs.Helper_Weekdays;

DROP TABLE IF EXISTS Kubs.Helper_Titles;

DROP TABLE IF EXISTS Kubs.Helper_ProductNames;
```

Listing 1: Skrypt SQL do usuwania obiektów bazy danych.

2 Zadanie 2 - Wymiar czasowy

Zadanie polegało na utworzeniu tabeli wymiaru czasu ‘DIM_TIME‘ oraz tabel pomocniczych (‘Helper_Months‘, ‘Helper_Weekdays‘) przechowujących polskie nazwy miesięcy i dni tygodnia. Tabela ‘DIM_TIME‘ zawiera klucz główny w formacie RRRRMMDD oraz rozbite atrybuty daty (rok, kwartał, miesiąc, dzień miesiąca, numer dnia tygodnia) oraz ich reprezentacje słowne uzyskane przez złączenie z tabelami pomocniczymi. Dane do tabeli ‘DIM_TIME‘ zostały wygenerowane na podstawie unikalnych dat (‘OrderDate‘, ‘ShipDate‘) z tabeli ‘Sales.SalesOrderHeader‘.

```
CREATE TABLE Kubs.Helper_Months (
    MonthNum INT PRIMARY KEY,
    MonthName_PL NVARCHAR(20) NOT NULL
);

INSERT INTO
    Kubs.Helper_Months (MonthNum, MonthName_PL)
VALUES
    (1, N'Styczeń'),
    (2, N'Luty'),
    (3, N'Marzec'),
    (4, N'Kwiecień'),
    (5, N'Maj'),
    (6, N'Czerwiec'),
    (7, N'Lipiec'),
    (8, N'Sierpień'),
    (9, N'Wrzesień'),
    (10, N'Październik'),
    (11, N'Listopad'),
    (12, N'Grudzień');

CREATE TABLE Kubs.Helper_Weekdays (
    WeekdayNum INT PRIMARY KEY,
    WeekdayName_PL NVARCHAR(20) NOT NULL
);

INSERT INTO
    Kubs.Helper_Weekdays (WeekdayNum, WeekdayName_PL)
VALUES
    (1, N'Niedziela'),
    (2, N'Poniedziałek'),
    (3, N'Wtorek'),
```

```
    (4, N'Środa'),
    (5, N'Czwartek'),
    (6, N'Piątek'),
    (7, N'Sobota');
```

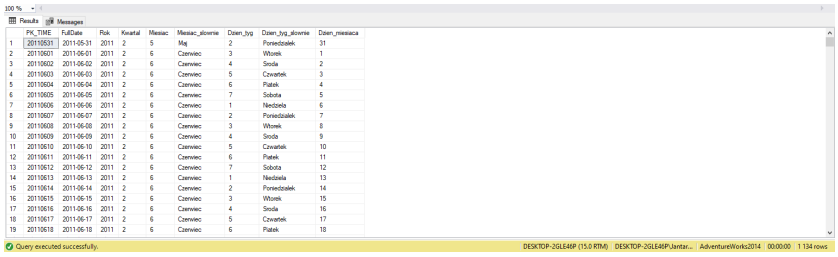
Listing 2: Tworzenie tabel pomocniczych dla wymiaru czasu.

```
CREATE TABLE Kubs.DIM_TIME (
    PK_TIME INT PRIMARY KEY,
    FullDate DATE NOT NULL,
    Rok INT NOT NULL,
    Kwartal INT NOT NULL,
    Miesiac INT NOT NULL,
    Miesiac_slownie NVARCHAR(20) NOT NULL,
    Dzien_tyg INT NOT NULL,
    Dzien_tyg_slownie NVARCHAR(20) NOT NULL,
    Dzien_miesiaca INT NOT NULL
);

WITH SourceDates AS (
    SELECT
        DISTINCT OrderDate AS CalendarDate
    FROM
        Sales.SalesOrderHeader
    WHERE
        OrderDate IS NOT NULL
    UNION
    SELECT
        DISTINCT ShipDate AS CalendarDate
    FROM
        Sales.SalesOrderHeader
    WHERE
        ShipDate IS NOT NULL
)
INSERT INTO
    Kubs.DIM_TIME (
        PK_TIME,
        FullDate,
        Rok,
        Kwartal,
        Miesiac,
        Miesiac_slownie,
        Dzien_tyg,
        Dzien_tyg_slownie,
```

```
38          Dzien_miesiaca
39      )
40  SELECT
41      (DATEPART(year, sd.CalendarDate) * 10000) + (DATEPART(month, sd.
        CalendarDate) * 100) + DATEPART(day, sd.CalendarDate) AS PK_TIME,
42      sd.CalendarDate AS FullDate,
43      DATEPART(year, sd.CalendarDate) AS Rok,
44      DATEPART(quarter, sd.CalendarDate) AS Kwartal,
45      DATEPART(month, sd.CalendarDate) AS Miesiac,
46      ISNULL(hm.MonthName_PL, 'Unknown') AS Miesiac_slownie,
47      DATEPART(weekday, sd.CalendarDate) AS Dzien_tyg,
48      ISNULL(hwd.WeekdayName_PL, 'Unknown') AS Dzien_tyg_slownie,
49      DATEPART(day, sd.CalendarDate) AS Dzien_miesiaca
50  FROM
51      SourceDates sd
52      LEFT JOIN Kubs.Helper_Months hm ON DATEPART(month, sd.
        CalendarDate) = hm.MonthNum
53      LEFT JOIN Kubs.Helper_Weekdays hwd ON DATEPART(weekday, sd.
        CalendarDate) = hwd.WeekdayNum;
```

Listing 3: Tworzenie i wypełnianie tabeli DIM_TIME.



Rysunek 1: Przykładowe dane w tabeli Kubs.DIM_TIME.

Przygotowanie DIM_TIME zostało zoptymalizowane dzięki wczesnej unii unikalnych ShipDate i OrderDate, przed procesem łączenia ich z tabelami pomocniczymi. Dzięki temu od początku operujemy na minimalnej liczbie rekordów - 1134.

3 Zadanie 3 - Elementarne czyszczenie danych

Celem tego zadania było zaimplementowanie podstawowego mechanizmu czyszczenia danych podczas procesu ładowania wymiarów. Polegało to na zastąpieniu wartości NULL w określonych kolumnach (‘Color‘, ‘SubCategoryName‘ w

‘DIM_PRODUCT‘; ‘CountryRegionCode‘, ‘Group‘ w ‘DIM_CUSTOMER‘ i ‘DIM_SALESPERSON‘) predefiniowanymi wartościami domyślnymi (’Unknown’, ’000’). Zastosowano funkcję ‘ISNULL()‘ podczas procesu INSERT danych.

```
    INSERT INTO
    Kubs.DIM_CUSTOMER (
        CustomerID,
        FirstName,
        LastName,
        Title,
        City,
        TerritoryName,
        CountryRegionCode,
        [Group]
    )
SELECT
    c.CustomerID,
    MIN(p.FirstName) AS FirstName,
    MIN(p.LastName) AS LastName,
    MIN(p.Title) AS Title,
    MIN(a.City) AS City,
    MIN(st.Name) AS TerritoryName,
    -- zmiana tutaj
    ISNULL(MIN(st.CountryRegionCode), '000') AS CountryRegionCode,
    ISNULL(MIN(st.[Group]), 'Unknown') AS [Group]
    -- zmiana tutaj
FROM
    Sales.Customer AS c
    LEFT JOIN Person.Person AS p ON c.PersonID = p.BusinessEntityID
    LEFT JOIN Sales.SalesTerritory AS st ON c.TerritoryID = st.
    TerritoryID
    LEFT JOIN Person.BusinessEntityAddress bea ON p.BusinessEntityID
     = bea.BusinessEntityID
    LEFT JOIN Person.Address AS a ON bea.AddressID = a.AddressID
WHERE
    c.PersonID IS NOT NULL
GROUP BY
    c.CustomerID;

INSERT INTO
    Kubs.DIM_PRODUCT (
        ProductID,
        Name,
```

```
        ListPrice,
        Color,
        SubCategoryName,
        CategoryName,
        Weight,
        Size
    )
SELECT
    DISTINCT p.ProductID,
    p.Name,
    p.ListPrice,
    -- zmiana tutaj
    ISNULL(p.Color, 'Unknown') AS Color,
    ISNULL(psc.Name, 'Unknown') AS SubCategoryName,
    -- zmiana tutaj
    pc.Name AS CategoryName,
    p.Weight,
    p.Size
FROM
    Production.Product AS p
    INNER JOIN Sales.SalesOrderDetail AS sod ON p.ProductID = sod.
    ProductID
    LEFT JOIN Production.ProductSubcategory AS psc ON p.
    ProductSubcategoryID = psc.ProductSubcategoryID
    LEFT JOIN Production.ProductCategory AS pc ON psc.
    ProductCategoryID = pc.ProductCategoryID;

INSERT INTO
    Kubs.DIM_SALESPERSON (
        SalesPersonID,
        FirstName,
        LastName,
        Title,
        Gender,
        CountryRegionCode,
        [Group]
    )
SELECT
    sp.BusinessEntityID AS SalesPersonID,
    p.FirstName,
    p.LastName,
    p.Title,
    e.Gender,
```

```
78      -- zmiana tutaj
79      ISNULL(st.CountryRegionCode, '000') AS CountryRegionCode,
80      ISNULL(st.[Group], 'Unknown') AS [Group]
81      -- zmiana tutaj
82  FROM
83      Sales.SalesPerson AS sp
84      INNER JOIN Person.Person AS p ON sp.BusinessEntityID = p.
        BusinessEntityID
85      INNER JOIN HumanResources.Employee AS e ON sp.BusinessEntityID =
         e.BusinessEntityID
86      LEFT JOIN Sales.SalesTerritory AS st ON sp.TerritoryID = st.
        TerritoryID;
87
88  INSERT INTO
89      Kubs.FACT_SALES (
90          ProductID,
91          CustomerID,
92          SalesPersonID,
93          OrderDate,
94          ShipDate,
95          OrderQty,
96          UnitPrice,
97          UnitPriceDiscount,
98          LineTotal
99      )
100 SELECT
101     sod.ProductID,
102     soh.CustomerID,
103     soh.SalesPersonID,
104     DATEPART(YEAR, soh.OrderDate) * 10000 + DATEPART(MONTH, soh.
        OrderDate) * 100 + DATEPART(DAY, soh.OrderDate) AS OrderDate,
105     DATEPART(YEAR, soh.ShipDate) * 10000 + DATEPART(MONTH, soh.
        ShipDate) * 100 + DATEPART(DAY, soh.ShipDate) AS ShipDate,
106     sod.OrderQty,
107     sod.UnitPrice,
108     sod.UnitPriceDiscount,
109     sod.LineTotal
110 FROM
111     Sales.SalesOrderDetail AS sod
112     INNER JOIN Sales.SalesOrderHeader AS soh ON sod.SalesOrderID =
        soh.SalesOrderID;
```

Listing 4: Przykład czyszczenia danych w instrukcji SELECT.

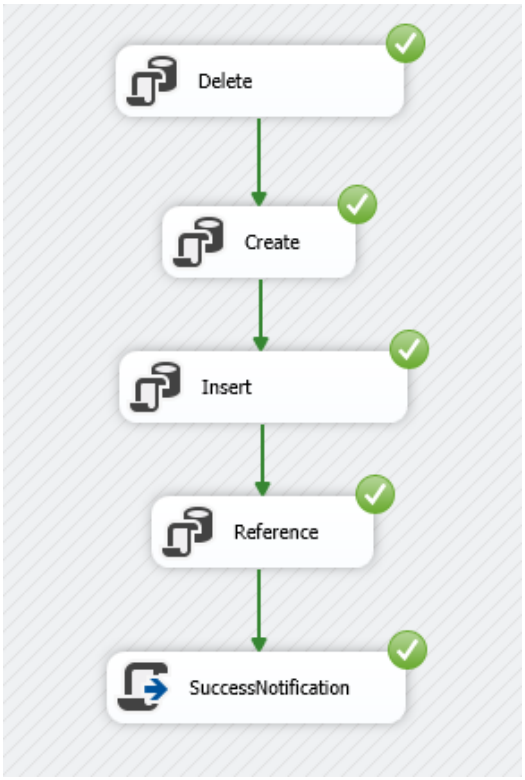
4 Zadanie 4 - Proces Extact - Transform - Load (SQL)

W tym zadaniu zautomatyzowano proces ETL z poprzednich kroków (oraz z Laboratorium 4) za pomocą pakietu SSIS (SQL Server Integration Services). Wykorzystano wyłącznie komponenty ‘Execute SQL Task‘ w zakładce ‘Control Flow‘ do wykonania kolejnych kroków:

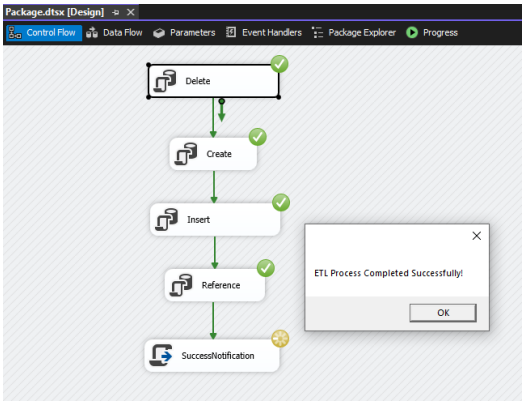
1. Usunięcie istniejących obiektów (skrypt z zadania 1).
2. Utworzenie struktur tabel wymiarów, faktów i pomocniczych (skrypt z zadania 2).
3. Wypełnienie tabel wymiarów (DIM_TIME, DIM_CUSTOMER, DIM_PRODUCT, DIM_SALESPERSON) wyczyszczonymi danymi.
4. Wypełnienie tabeli faktów (FACT_SALES) danymi zagregowanymi i kluczami obcymi.
5. Dodanie więzów integralności.
6. Dodano obsługę błędów (Event Handlers)
7. Dodano powiadomienie o sukcesie (skrypt C#).

```
1  -- poprzednie wiersze z poprzedniej listy...
2  ALTER TABLE
3      Kubs.FACT_SALES WITH NOCHECK
4  ADD
5      CONSTRAINT FK_FACT_SALES_DIM_TIME_OrderDate FOREIGN KEY (
       OrderDate) REFERENCES Kubs.DIM_TIME(PK_TIME);
6
7  GO
8  ALTER TABLE
9      Kubs.FACT_SALES WITH NOCHECK
10 ADD
11     CONSTRAINT FK_FACT_SALES_DIM_TIME_ShipDate FOREIGN KEY (ShipDate
       ) REFERENCES Kubs.DIM_TIME(PK_TIME);
```

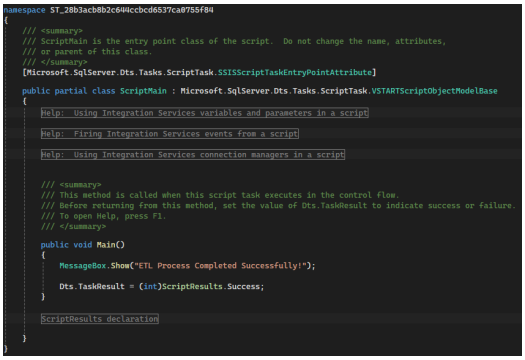
Listing 5: Dodanie nowych CONSTRAINT



Rysunek 2: Schemat Control Flow pakietu SSIS realizującego ETL za pomocą Execute SQL Task.



Rysunek 3: Komunikat oznajmiający sukces załadowania danych.

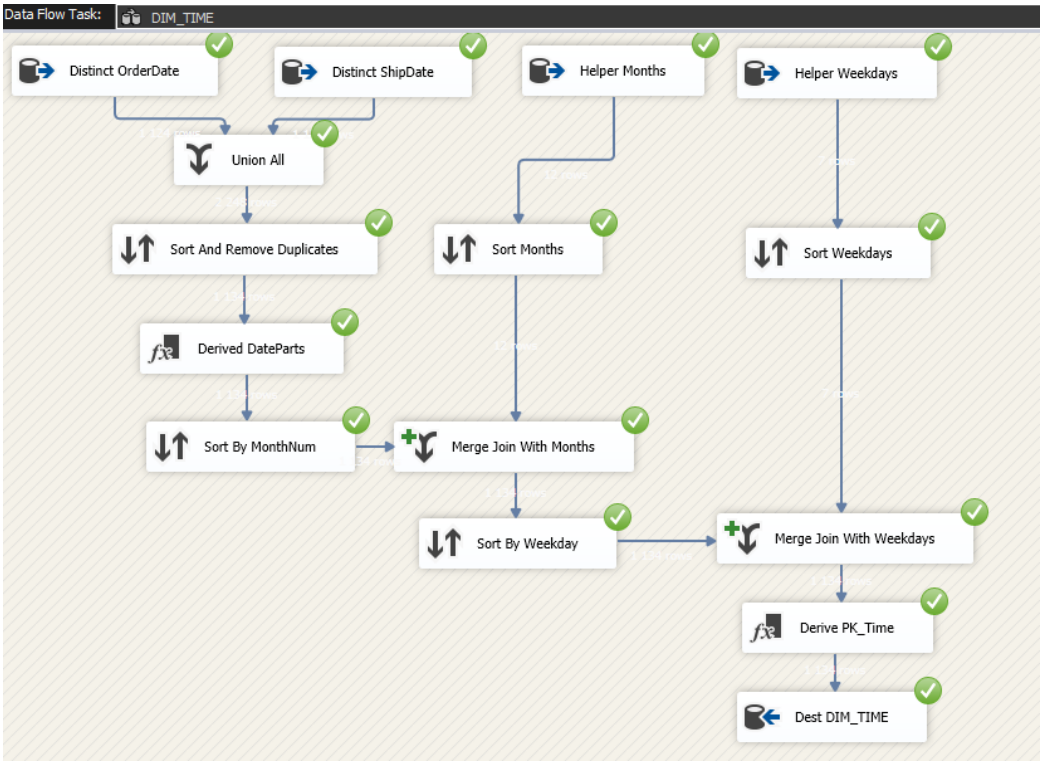


Rysunek 4: Skrypt odpowiedzialny za oznajmienie sukcesu.

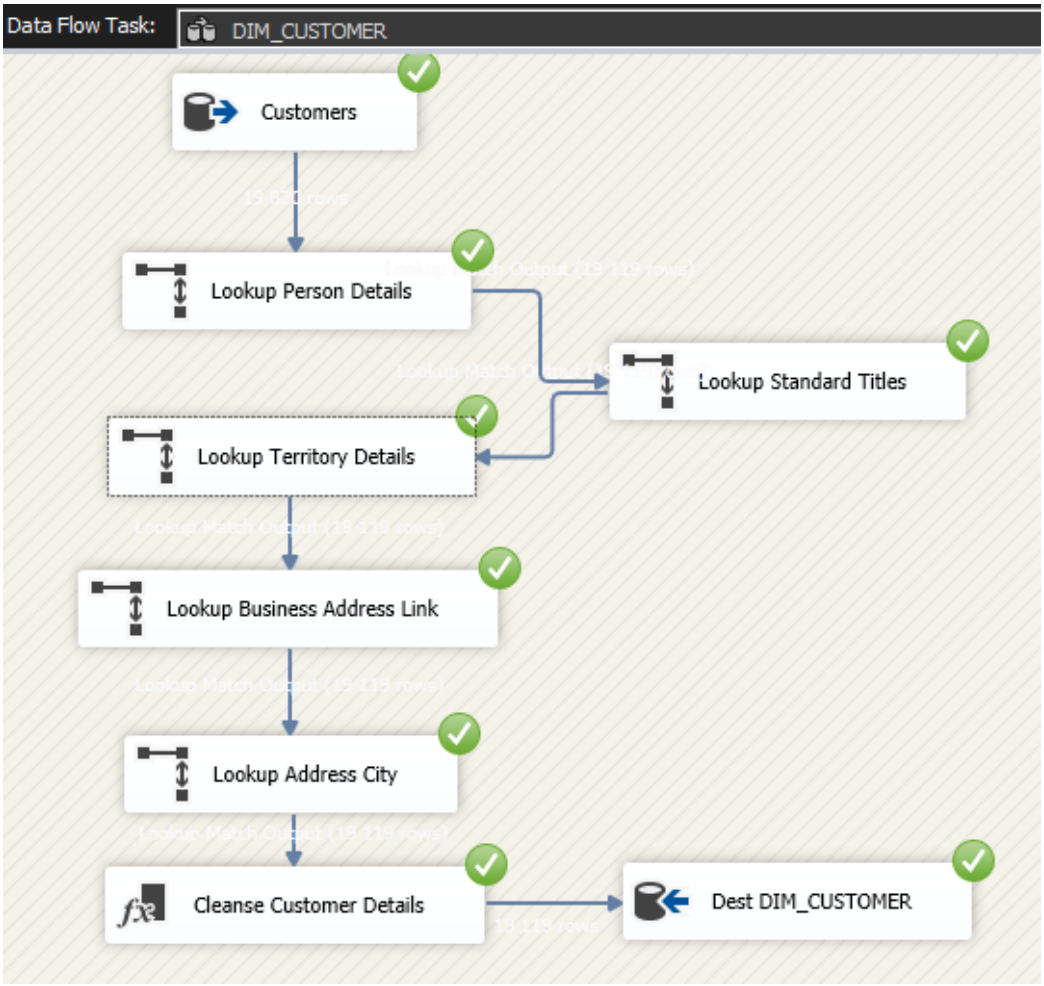
5 Zadanie 5 - ETL (prawie) bez SQLa (Data Flow)

Zadanie 5 polegało na refaktoryzacji procesu ETL z zadania 4, tak aby ładowanie danych do wymiaru czasu (‘DIM_TIME‘) oraz co najmniej jednego innego wybranego wymiaru odbywało się przy użyciu komponentów graficznych z zakładki ‘Data Flow‘ w SSIS, minimalizując użycie bezpośrednich zapytań SQL w tych krokach. Wykorzystano m.in.:

- ‘OLE DB Source‘: Do pobierania danych ze źródłowych tabel AdventureWorks.
- ‘Derived Column‘: Do tworzenia nowych kolumn (np. ‘PK_TIME‘, ‘Rok‘, ‘Kwartał‘), implementacji czyszczenia danych (‘ISNULL(Color, ’Unknown’)‘). * ‘Union All‘: Do połączenia danych (np. OrderDate i ShipDate).
- ‘Lookup‘: Do dołączania informacji z innych tabel. * ‘Sort‘: Do sortowania danych, a w szczególności do usuwania duplikatów (np. dla ‘DIM_CUSTOMER‘ w celu zapewnienia unikalności ‘CustomerID‘). * ‘Fuzzy Grouping‘ / ‘Fuzzy Lookup‘: fuzzy lookup do ujednolicenia nazw produktów.
- ‘OLE DB Destination‘: Do zapisywania przetworzonych danych do docelowych tabel ‘Kubs‘.



Rysunek 5: Schemat Data Flow Task dla ładowania wymiaru czasu (DIM_TIME).



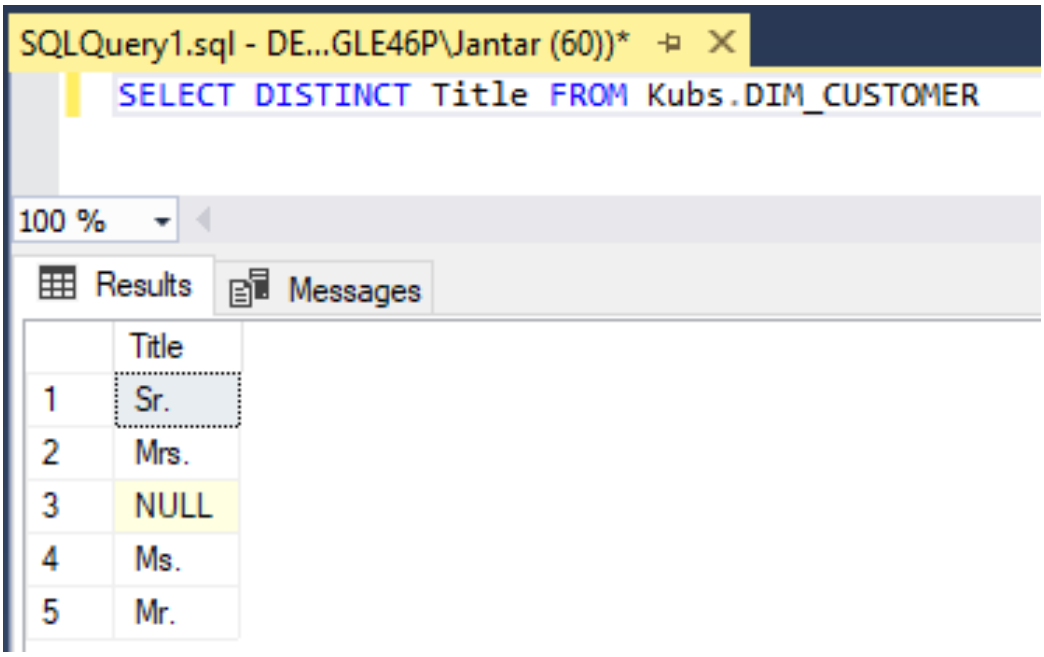
Rysunek 6: Schemat Data Flow Task dla DIM_CUSTOMER.

W powyższym schemacie dzięki wyczyszczeniu danych przy użyciu kolejnej helper tablicy można było ujednolicić tytuły klientów.

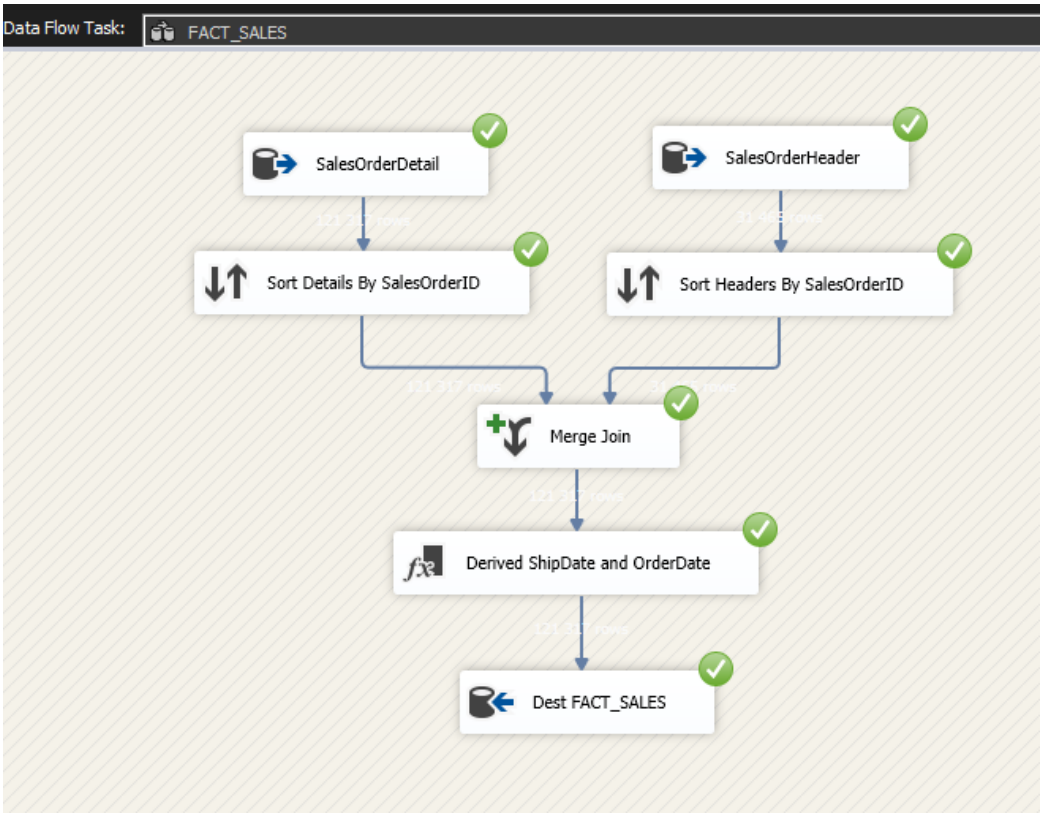
```
CREATE TABLE Kubs.Helper_Titles (
    SourceTitle NVARCHAR(8) PRIMARY KEY,
    StandardTitle NVARCHAR(8) NOT NULL
);

INSERT INTO
    Kubs.Helper_Titles (SourceTitle, StandardTitle)
VALUES
    ('Mr.', 'Mr.'),
    ('Ms', 'Ms.'),
```

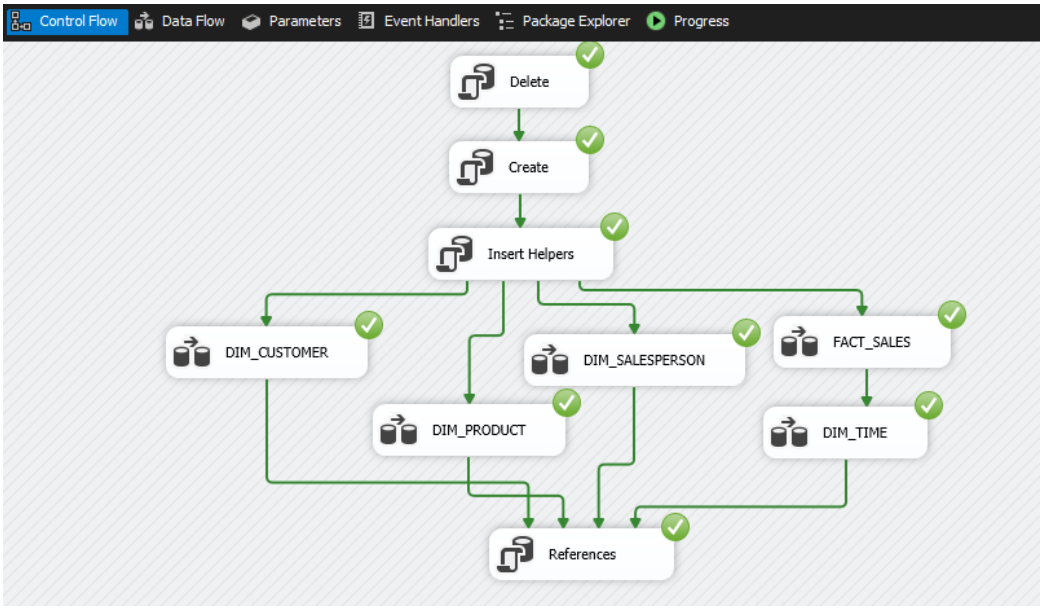
```
11      ('Ms.', 'Ms.'),
12      ('Mrs.', 'Mrs.'),
13      ('Sra', 'Sr.'),
14      ('Sr.', 'Sr.');
```



Rysunek 7: Ujednolicone tytuły klientów w DIM_CUSTOMER.

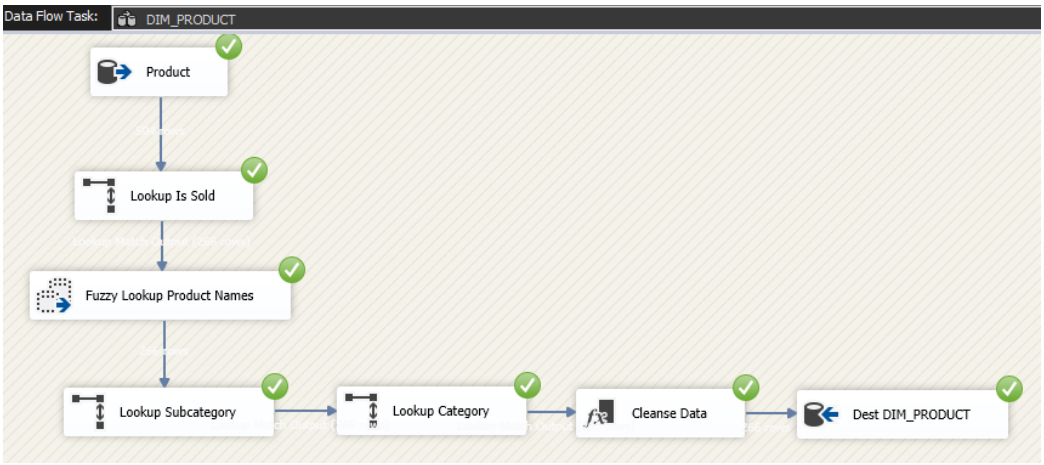


Rysunek 8: Schemat Data Flow Task dla FACT_SALES.

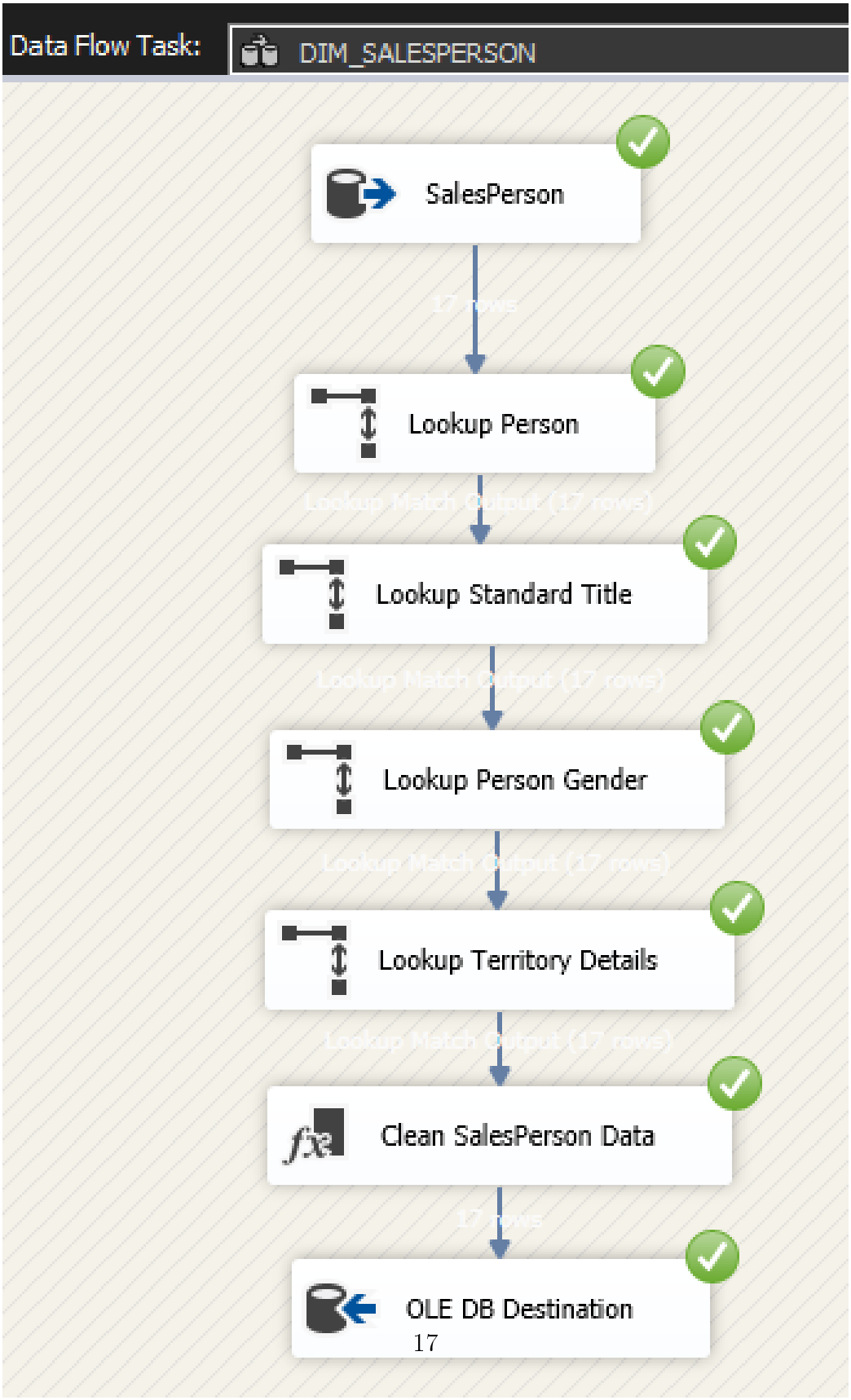


Rysunek 9: Schemat Control Flow pakietu SSIS wykorzystującego Data Flow Task.

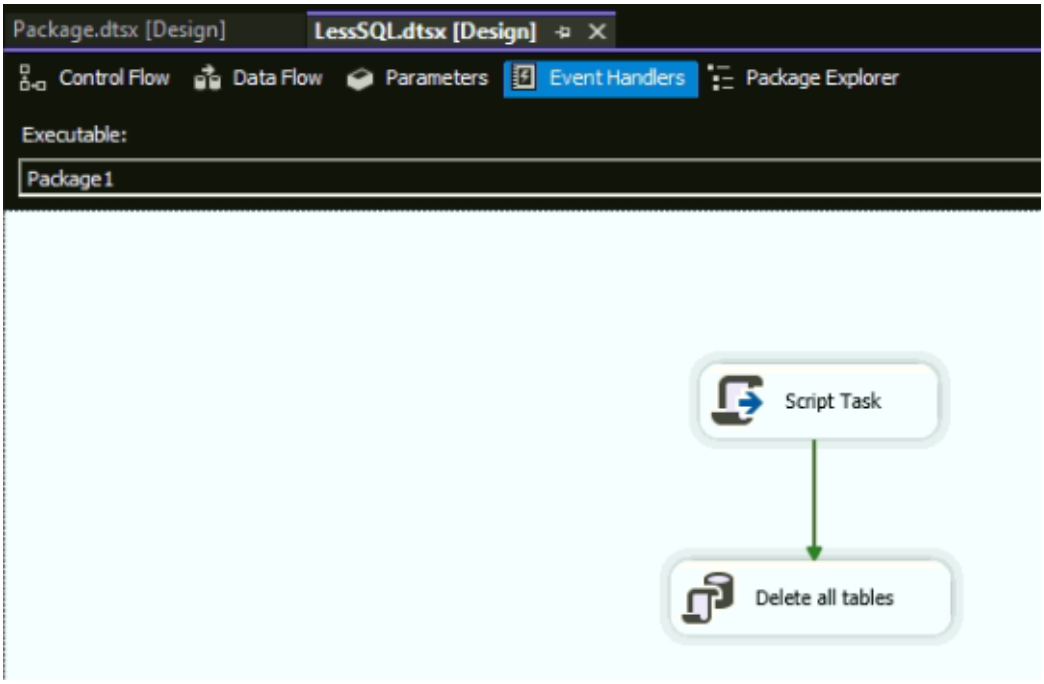
W celu lepszego zaznajomienia się z procesem wykonano również Data Flow Taski dla pozostałych wymiarów.



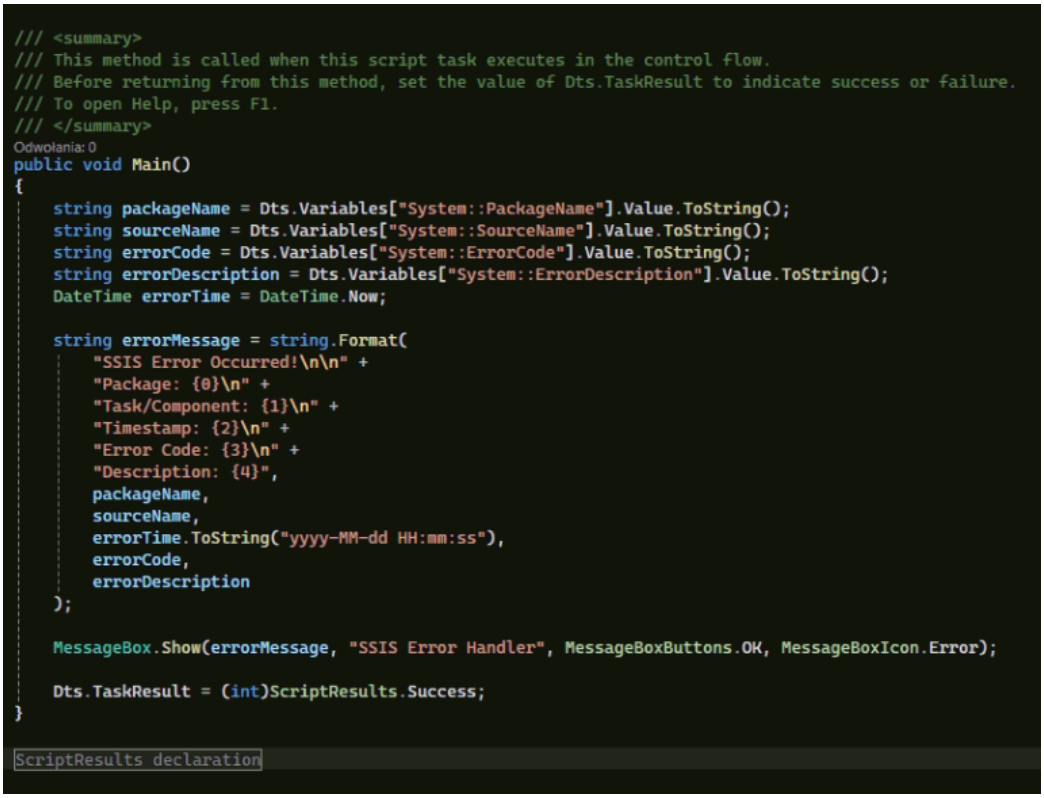
Rysunek 10: Schemat Data Flow Task dla DIM_PRODUCT.



Rysunek 11: Schemat Data Flow Task dla DIM_SALESPERSON.



Rysunek 12: Schemat Error Handler.



Rysunek 13: Kod skryptu Error Handler.

6 Wnioski

Jednym z największych plusów takiego procesu ETL jest jego powtarzalność. Nieważne, jakie błędy wyszły, pierwszym krokiem jest usunięcie wszystkich tabel i stworzenie ich jeszcze raz. Ułatwiło to pracę, testowanie i zwiększyło pewność, że proces działa za każdym razem.

Wizualne Data Flow Taski faktycznie są bardziej przejrzyste i jasne od SQL, zwłaszcza dla osób mniej technicznych. Z drugiej strony, dla osoby dobrze znającej SQL kwerendy dobrze rozbite na ETL równiez mogą być czytelne i dużo szybsze do napisania - GUI SSIS wymaga dużo klikania.

Uważam jednak, że jeszcze lepszym podejściem jest takie, które jest np. w Power Query - te taski definiuje się w języku programowania M, ale nadal jasno widać, który krok co robi dzięki jego nazwie. Kroki można tam również szybko dodawać i zmieniać dzięki GUI, a także zmieniać kolejność.

W SSIS każdy proces był czasochłonny, pod względem User Experience

nienajlepszy. Czasem występowały błędy, które naprawiał tylko restart Visual Studio.

Fuzzy Lookup został użyty w procesie wypełniania DIM_PRODUCT. Praktycznie ten sam produkt był definiowany parę razy dla różnych kolorów lub rozmiarów. W moim eksperymencie Fuzzy Lookup miał wyeliminować tylko różne rozmiary w nazwach (np. "HL Road Frame - Red, 62"i "HL Road Frame - Red, 44") i tak też się udało, dzięki użyciu threshold na poziomie 66%. Był to tylko eksperyment, i lepiej do tego nadawałby się Fuzzy Grouping - teraz mamy wiele produktów o tej samej nazwie, ale teoretycznie są to różne wiersze w tabeli. Z drugiej strony, w tabeli są kolumny dotyczące wagi, rozmiaru i koloru produktów, przez co takie grupowanie by pozbyło się szczegółów.

Stworzenie dedykowanego wymiaru czasu (DIM_TIME) jest fundamentalnym krokiem w budowie hurtowni. Dostarcza on spójnego zestawu atrybutów (rok, kwartał, miesiąc, dzień tygodnia etc.) powiązanych z kluczem głównym (np. RRRRMMDD), co jest kluczowe dla późniejszych analiz i raportowania, umożliwiając łatwe agregowanie danych i drillowanie ich w kontekście czasowym.

Podsumowując, laboratorium pokazało praktyczne aspekty tworzenia wymiaru czasu i implementacji czyszczenia danych w procesie ETL, a także naświetliło różnice w podejściach do automatyzacji tego procesu w SSIS. Podejście Data Flow oferuje większą wizualną przejrzystość kosztem większego nakładu pracy konfiguracyjnej i potencjalnych problemów z narzędziem, podczas gdy podejście oparte na SQL może być szybsze dla znających język, ale trudniejsze w utrzymaniu przy dużej złożoności.