

# 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'.

```
1 IF EXISTS (  
2     SELECT  
3         *  
4     FROM  
5         INFORMATION_SCHEMA.TABLE_CONSTRAINTS  
6     WHERE  
7         CONSTRAINT_SCHEMA = 'Kubs'  
8         AND CONSTRAINT_NAME = 'FK_FACT_SALES_DIM_CUSTOMER'  
9         AND TABLE_NAME = 'FACT_SALES'  
10 )  
11 ALTER TABLE  
12     Kubs.FACT_SALES DROP CONSTRAINT FK_FACT_SALES_DIM_CUSTOMER;  
13  
14 IF EXISTS (  
15     SELECT  
16         *  
17     FROM  
18         INFORMATION_SCHEMA.TABLE_CONSTRAINTS  
19     WHERE
```

```

20         CONSTRAINT_SCHEMA = 'Kubs'
21         AND CONSTRAINT_NAME = 'FK_FACT_SALES_DIM_PRODUCT'
22         AND TABLE_NAME = 'FACT_SALES'
23     )
24 ALTER TABLE
25     Kubs.FACT_SALES DROP CONSTRAINT FK_FACT_SALES_DIM_PRODUCT;
26
27 IF EXISTS (
28     SELECT
29         *
30     FROM
31         INFORMATION_SCHEMA.TABLE_CONSTRAINTS
32     WHERE
33         CONSTRAINT_SCHEMA = 'Kubs'
34         AND CONSTRAINT_NAME = 'FK_FACT_SALES_DIM SALESPERSON'
35         AND TABLE_NAME = 'FACT_SALES'
36 )
37 ALTER TABLE
38     Kubs.FACT_SALES DROP CONSTRAINT FK_FACT_SALES_DIM SALESPERSON;
39
40 DROP TABLE IF EXISTS Kubs.FACT_SALES;
41
42 DROP TABLE IF EXISTS Kubs.DIM_CUSTOMER;
43
44 DROP TABLE IF EXISTS Kubs.DIM_PRODUCT;
45
46 DROP TABLE IF EXISTS Kubs.DIM SALESPERSON;
47
48 DROP TABLE IF EXISTS Kubs.DIM_TIME;
49
50 DROP TABLE IF EXISTS Kubs.Helper_Months;
51
52 DROP TABLE IF EXISTS Kubs.Helper_Weekdays;
53
54 DROP TABLE IF EXISTS Kubs.Helper_Titles;
55
56 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'.

```
1 CREATE TABLE Kubs.Helper_Months (  
2     MonthNum INT PRIMARY KEY,  
3     MonthName_PL NVARCHAR(20) NOT NULL  
4 );  
5  
6 INSERT INTO  
7     Kubs.Helper_Months (MonthNum, MonthName_PL)  
8 VALUES  
9     (1, N'Styczeń'),  
10    (2, N'Luty'),  
11    (3, N'Marzec'),  
12    (4, N'Kwiecień'),  
13    (5, N'Maj'),  
14    (6, N'Czerwiec'),  
15    (7, N'Lipiec'),  
16    (8, N'Sierpień'),  
17    (9, N'Wrzesień'),  
18    (10, N'Październik'),  
19    (11, N'Listopad'),  
20    (12, N'Grudzień');  
21  
22 CREATE TABLE Kubs.Helper_Weekdays (  
23     WeekdayNum INT PRIMARY KEY,  
24     WeekdayName_PL NVARCHAR(20) NOT NULL  
25 );  
26  
27 INSERT INTO  
28     Kubs.Helper_Weekdays (WeekdayNum, WeekdayName_PL)  
29 VALUES  
30     (1, N'Niedziela'),  
31     (2, N'Poniedziałek'),  
32     (3, N'Wtorek'),
```

```

33      (4, N'Środa'),
34      (5, N'Czwartek'),
35      (6, N'Piątek'),
36      (7, N'Sobota');

```

Listing 2: Tworzenie tabel pomocniczych dla wymiaru czasu.

```

1  CREATE TABLE Kubs.DIM_TIME (
2      PK_TIME INT PRIMARY KEY,
3      FullDate DATE NOT NULL,
4      Rok INT NOT NULL,
5      Kwartal INT NOT NULL,
6      Miesiac INT NOT NULL,
7      Miesiac_slownie NVARCHAR(20) NOT NULL,
8      Dzień_tyg INT NOT NULL,
9      Dzień_tyg_slownie NVARCHAR(20) NOT NULL,
10     Dzień_miesiaca INT NOT NULL
11 );
12
13 WITH SourceDates AS (
14     SELECT
15         DISTINCT OrderDate AS CalendarDate
16     FROM
17         Sales.SalesOrderHeader
18     WHERE
19         OrderDate IS NOT NULL
20     UNION
21     SELECT
22         DISTINCT ShipDate AS CalendarDate
23     FROM
24         Sales.SalesOrderHeader
25     WHERE
26         ShipDate IS NOT NULL
27 )
28 INSERT INTO
29     Kubs.DIM_TIME (
30         PK_TIME,
31         FullDate,
32         Rok,
33         Kwartal,
34         Miesiac,
35         Miesiac_slownie,
36         Dzień_tyg,
37         Dzień_tyg_slownie,

```

```

38         Dzień_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 Kwartał,
45     DATEPART(month, sd.CalendarDate) AS Miesiąc,
46     ISNULL(hm.MonthName_PL, 'Unknown') AS Miesiąc_słownie,
47     DATEPART(weekday, sd.CalendarDate) AS Dzień_tyg,
48     ISNULL(hwd.WeekdayName_PL, 'Unknown') AS Dzień_tyg_słownie,
49     DATEPART(day, sd.CalendarDate) AS Dzień_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.

	PK_TIME	FullDate	Rok	Kwartał	Miesiąc	Miesiąc_słownie	Dzień_tyg	Dzień_tyg_słownie	Dzień_miesiaca
1	20110531	2011-05-31	2011	2	5	Maj	2	Poniedziałek	31
2	20110601	2011-06-01	2011	2	6	Czerwiec	3	Wtorek	1
3	20110602	2011-06-02	2011	2	6	Czerwiec	4	Środa	2
4	20110603	2011-06-03	2011	2	6	Czerwiec	5	Czwartek	3
5	20110604	2011-06-04	2011	2	6	Czerwiec	6	Piątek	4
6	20110605	2011-06-05	2011	2	6	Czerwiec	7	Śobota	5
7	20110606	2011-06-06	2011	2	6	Czerwiec	1	Niedziela	6
8	20110607	2011-06-07	2011	2	6	Czerwiec	2	Poniedziałek	7
9	20110608	2011-06-08	2011	2	6	Czerwiec	3	Wtorek	8
10	20110609	2011-06-09	2011	2	6	Czerwiec	4	Środa	9
11	20110610	2011-06-10	2011	2	6	Czerwiec	5	Czwartek	10
12	20110611	2011-06-11	2011	2	6	Czerwiec	6	Piątek	11
13	20110612	2011-06-12	2011	2	6	Czerwiec	7	Śobota	12
14	20110613	2011-06-13	2011	2	6	Czerwiec	1	Niedziela	13
15	20110614	2011-06-14	2011	2	6	Czerwiec	2	Poniedziałek	14
16	20110615	2011-06-15	2011	2	6	Czerwiec	3	Wtorek	15
17	20110616	2011-06-16	2011	2	6	Czerwiec	4	Środa	16
18	20110617	2011-06-17	2011	2	6	Czerwiec	5	Czwartek	17
19	20110618	2011-06-18	2011	2	6	Czerwiec	6	Piątek	18

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.

```

1      INSERT INTO
2      Kubs.DIM_CUSTOMER (
3          CustomerID,
4          FirstName,
5          LastName,
6          Title,
7          City,
8          TerritoryName,
9          CountryRegionCode,
10         [Group]
11     )
12 SELECT
13     c.CustomerID,
14     MIN(p.FirstName) AS FirstName,
15     MIN(p.LastName) AS LastName,
16     MIN(p.Title) AS Title,
17     MIN(a.City) AS City,
18     MIN(st.Name) AS TerritoryName,
19     -- zmiana tutaj
20     ISNULL(MIN(st.CountryRegionCode), '000') AS CountryRegionCode,
21     ISNULL(MIN(st.[Group]), 'Unknown') AS [Group]
22     -- zmiana tutaj
23 FROM
24     Sales.Customer AS c
25     LEFT JOIN Person.Person AS p ON c.PersonID = p.BusinessEntityID
26     LEFT JOIN Sales.SalesTerritory AS st ON c.TerritoryID = st.
27     TerritoryID
28     LEFT JOIN Person.BusinessEntityAddress bea ON p.BusinessEntityID
29     = bea.BusinessEntityID
30     LEFT JOIN Person.Address AS a ON bea.AddressID = a.AddressID
31 WHERE
32     c.PersonID IS NOT NULL
33 GROUP BY
34     c.CustomerID;
35
36 INSERT INTO
37     Kubs.DIM_PRODUCT (
38         ProductID,
39         Name,

```

```

38         ListPrice,
39         Color,
40         SubCategoryName,
41         CategoryName,
42         Weight,
43         Size
44     )
45 SELECT
46     DISTINCT p.ProductID,
47     p.Name,
48     p.ListPrice,
49     -- zmiana tutaj
50     ISNULL(p.Color, 'Unknown') AS Color,
51     ISNULL(psc.Name, 'Unknown') AS SubCategoryName,
52     -- zmiana tutaj
53     pc.Name AS CategoryName,
54     p.Weight,
55     p.Size
56 FROM
57     Production.Product AS p
58     INNER JOIN Sales.SalesOrderDetail AS sod ON p.ProductID = sod.
59     ProductID
60     LEFT JOIN Production.ProductSubcategory AS psc ON p.
61     ProductSubcategoryID = psc.ProductSubcategoryID
62     LEFT JOIN Production.ProductCategory AS pc ON psc.
63     ProductCategoryID = pc.ProductCategoryID;
64
65 INSERT INTO
66     Kubs.DIM_SALESPERSON (
67         SalesPersonID,
68         FirstName,
69         LastName,
70         Title,
71         Gender,
72         CountryRegionCode,
73         [Group]
74     )
75 SELECT
76     sp.BusinessEntityID AS SalesPersonID,
77     p.FirstName,
78     p.LastName,
79     p.Title,
80     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.



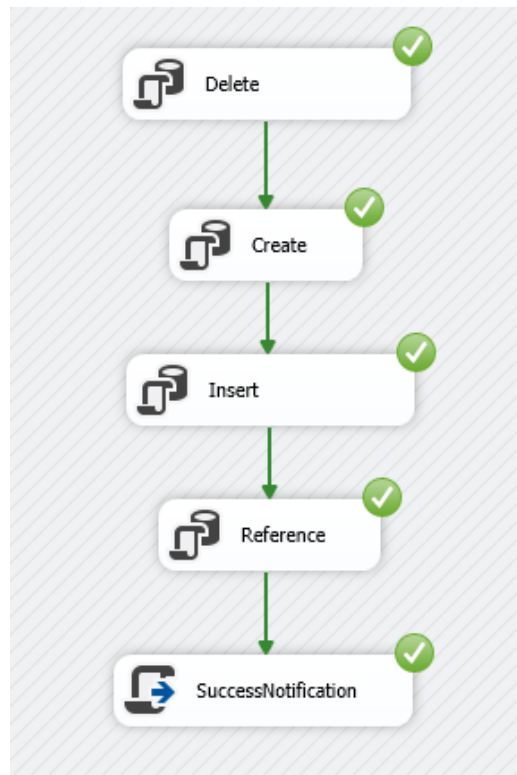
## 4 Zadanie 4 - Proces Extact - Transform - Load (SQL)

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:

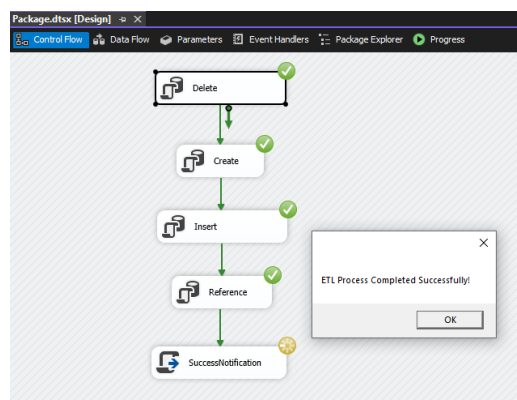
1. Usunięcie istniejących obiektów (skrypt z zadania 1).
2. Utworzenie struktur tabel wymiarów, faktów i pomocniczych (skrypt z zadania 2).
3. Wypełnienie tabel wymiarów (DIM\_TIME, DIM\_CUSTOMER, DIM\_PRODUCT, DIM SALESPERSON) wyczyszczonymi danymi.
4. Wypełnienie tabeli faktów (FACT\_SALES) danymi 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 (
6          OrderDate) REFERENCES Kubs.DIM_TIME(PK_TIME);
7
8  GO
9  ALTER TABLE
10     Kubs.FACT_SALES WITH NOCHECK
11  ADD
12     CONSTRAINT FK_FACT_SALES_DIM_TIME_ShipDate FOREIGN KEY (ShipDate
13         ) 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.

```

//spacespace ST_28b3ac88b2c64ccbc6537ca9755f84
{
    /// <summary>
    /// ScriptMain is the entry point class of the script. Do not change the name, attributes,
    /// or parent of this class.
    /// </summary>
    [Microsoft.SqlServer.Dts.Tasks.ScriptTask.SSISScriptTaskEntryPointAttribute]
    public partial class ScriptMain : Microsoft.SqlServer.Dts.Tasks.ScriptTask.VSTARTScriptObjectModelBase
    {
        Help: Using Integration Services variables and parameters in a script
        Help: Firing Integration Services events from a script
        Help: Using Integration Services connection managers in a script

        /// <summary>
        /// This method is called when this script task executes in the control flow.
        /// Before returning from this method, set the value of Dts.TaskResult to indicate success or failure.
        /// To open Help, press F1.
        /// </summary>
        public void Main()
        {
            MessageBox.Show("ETL Process Completed Successfully!");
            Dts.TaskResult = (int)ScriptResults.Success;
        }
    }
}

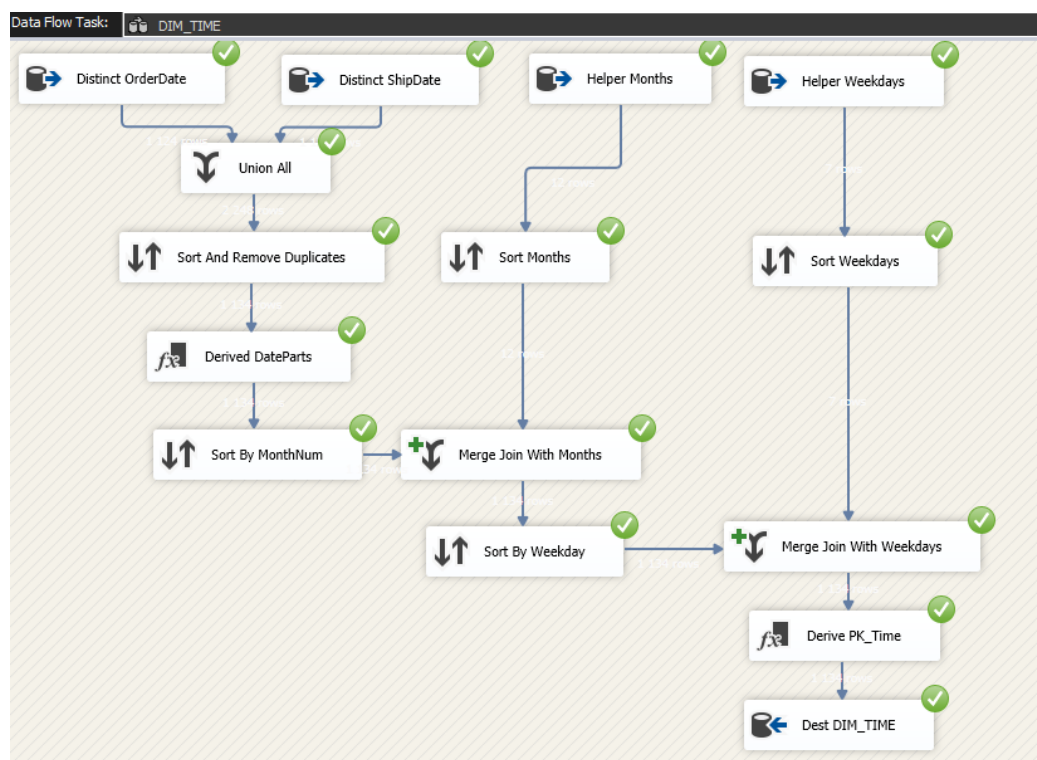
```

Rysunek 4: Skrypt odpowiedzialny za oznajmienie sukcesu.

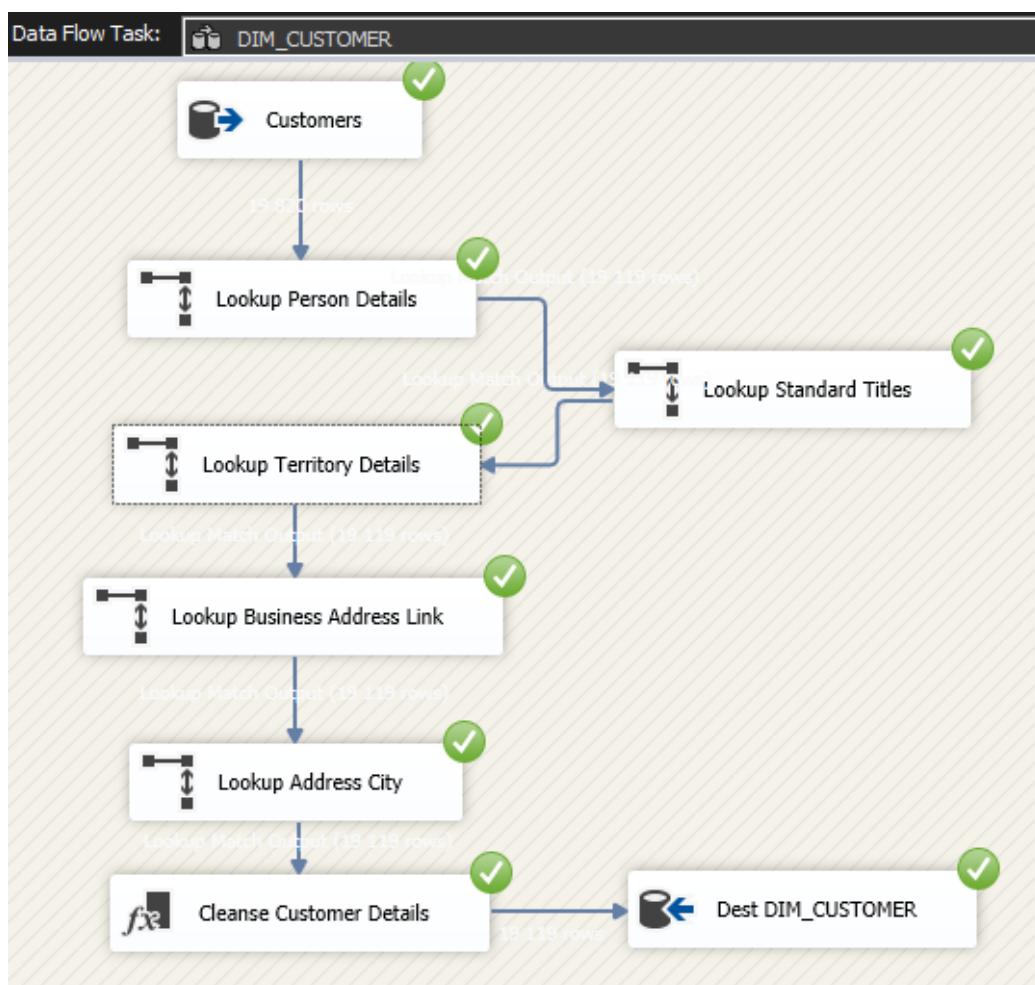
## 5 Zadanie 5 - ETL (prawie) bez SQLa (Data Flow)

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.

```

1 CREATE TABLE Kubs.Helper_Titles (
2     SourceTitle NVARCHAR(8) PRIMARY KEY,
3     StandardTitle NVARCHAR(8) NOT NULL
4 );
5
6 INSERT INTO
7     Kubs.Helper_Titles (SourceTitle, StandardTitle)
8 VALUES
9     ('Mr.', 'Mr.'),
10    ('Ms', 'Ms.'),

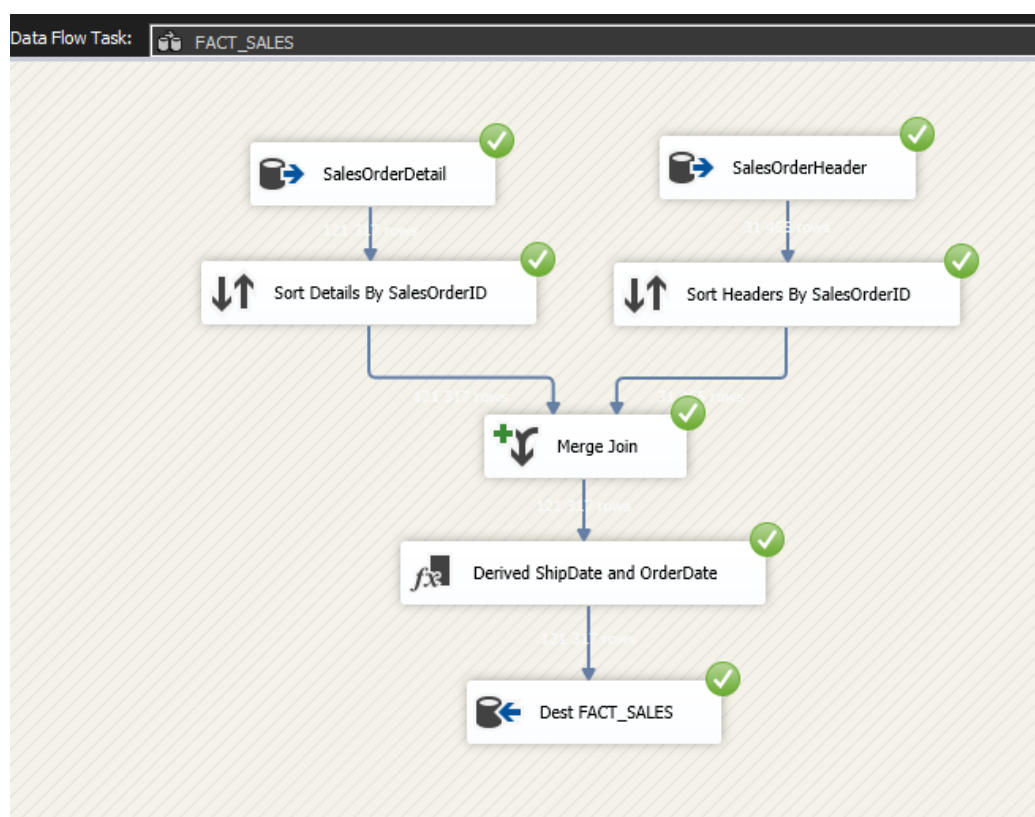
```

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

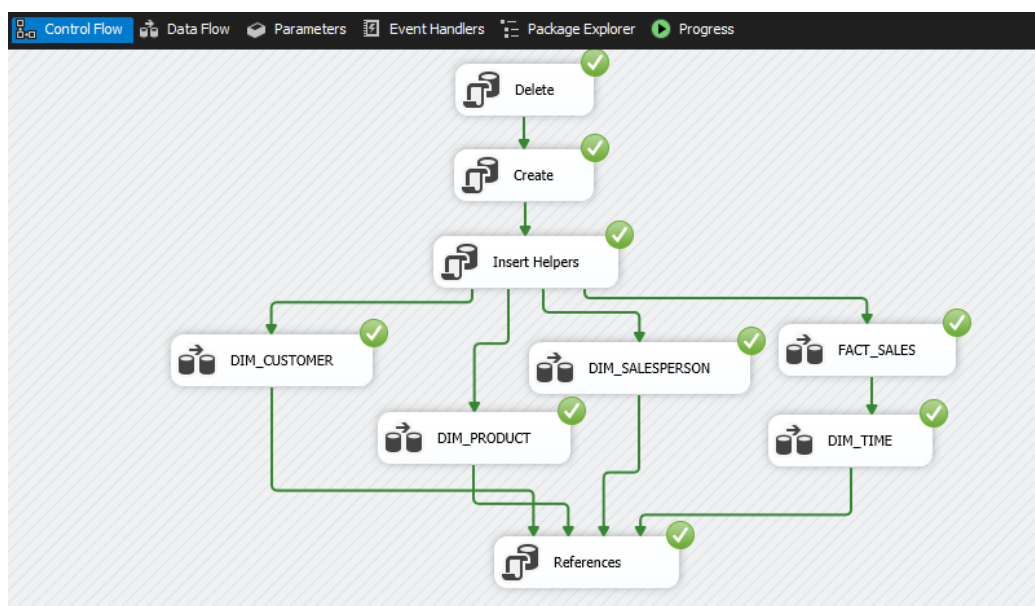
The screenshot shows a SQL query window titled "SQLQuery1.sql - DE...GLE46P\Jantar (60))\*" with the query: `SELECT DISTINCT Title FROM Kubs.DIM_CUSTOMER`. Below the query, the "Results" tab is active, displaying a table with 5 rows and 1 column titled "Title". The rows contain the values: "Sr.", "Mrs.", "NULL", "Ms.", and "Mr.". The "Messages" tab is also visible but empty.

	Title
1	Sr.
2	Mrs.
3	NULL
4	Ms.
5	Mr.

Rysunek 7: Ujednolicone tytuły klientów w DIM\_CUSTOMER.

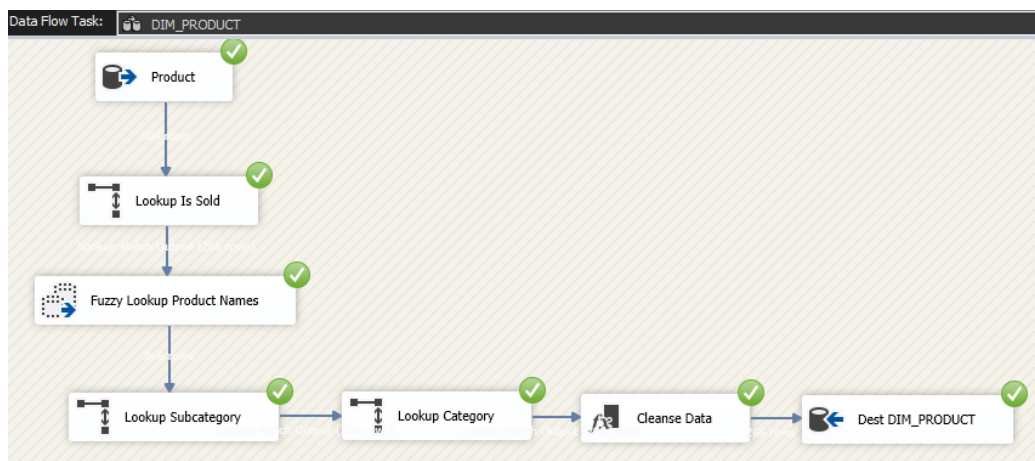


Rysunek 8: Schemat Data Flow Task dla FACT\_SALES.



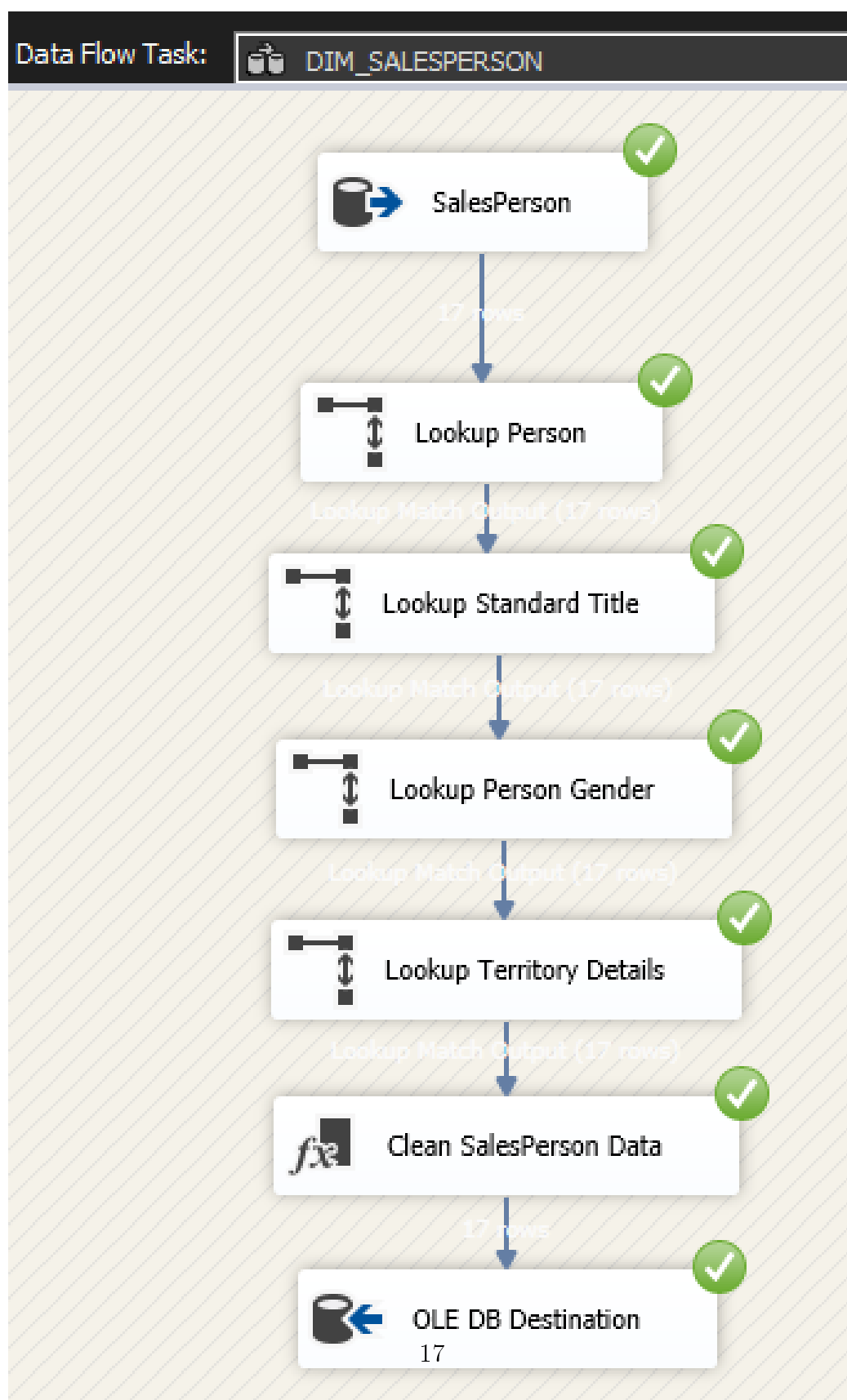
Rysunek 9: Schemat Control Flow pakietu SSIS wykorzystującego Data Flow Task.

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

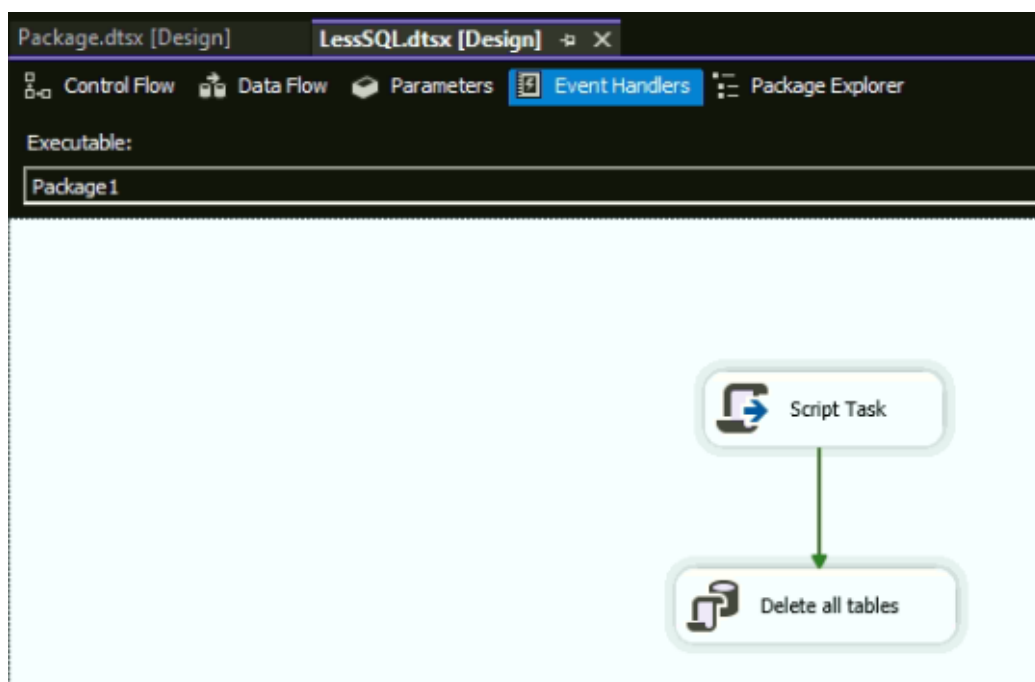


Rysunek 10: Schemat Data Flow Task dla DIM\_PRODUCT.





Rysunek 11: Schemat Data Flow Task dla DIM\_SALESPERSON.



Rysunek 12: Schemat Error Handler.

```

/// <summary>
/// This method is called when this script task executes in the control flow.
/// Before returning from this method, set the value of Dts.TaskResult to indicate success or failure.
/// To open Help, press F1.
/// </summary>
Odwolania: 0
public void Main()
{
    string packageName = Dts.Variables["System::PackageName"].Value.ToString();
    string sourceName = Dts.Variables["System::SourceName"].Value.ToString();
    string errorCode = Dts.Variables["System::ErrorCode"].Value.ToString();
    string errorDescription = Dts.Variables["System::ErrorDescription"].Value.ToString();
    DateTime errorTime = DateTime.Now;

    string errorMessage = string.Format(
        "SSIS Error Occurred!\n\n" +
        "Package: {0}\n" +
        "Task/Component: {1}\n" +
        "Timestamp: {2}\n" +
        "Error Code: {3}\n" +
        "Description: {4}",
        packageName,
        sourceName,
        errorTime.ToString("yyyy-MM-dd HH:mm:ss"),
        errorCode,
        errorDescription
    );

    MessageBox.Show(errorMessage, "SSIS Error Handler", MessageBoxButtons.OK, MessageBoxIcon.Error);

    Dts.TaskResult = (int)ScriptResults.Success;
}
ScriptResults declaration

```

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 również 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.