**Hurtownie danych – Ćwiczenie 5.**

PWr. WIZ, Informatyka, Data: 31.03.2020

| Student | Email: 242493@student.pwr.edu.pl | Ocena |
|---|---|---|
| Indeks | 242493 | |
| Imię | Arkadiusz | |
| Nazwisko | Rasz | |

# Spis treści

## Zad. 1. Usunięcie tabel

Przygotować instrukcję usuwającą każdą z tabel utworzonych w trakcie pracy nad listą 4.

```
drop table if exists [Rasz].[Fact_Sales];

drop table if exists [Rasz].[Dim_Time];

drop table if exists [Rasz].[Dim_Salesperson];

drop table if exists [Rasz].[Dim_Customer];

drop table if exists [Rasz].[Dim_Product];
```

## Zad. 2. Wymiar czasowy

Przygotować wymiar czasowy: utworzyć i wypełnić danymi tabelę DIM_TIME. Tabela DIM_TIME powinna być tabelą zawierającą wymiar czasowy (klucze obce do tej tabeli znajdują się w tabeli faktów).

### Tworzenie tabeli

```
CREATE TABLE [Rasz].[Dim_Time] (
	[Time] INT NOT NULL PRIMARY KEY,
	[Year] SMALLINT NOT NULL,
	[Quarter] TINYINT NOT NULL,
	[Month] TINYINT NOT NULL,
	[MonthWord] NVARCHAR(10) NOT NULL,
	[Weekday] NVARCHAR(10) NOT NULL,
	[DayInMonth] TINYINT NOT NULL
);
```

### Dodanie kluczy obcych

```
ALTER TABLE [Rasz].[Fact_Sales]
ADD CONSTRAINT FK_FactSales_OrderDate_DimTime_Time FOREIGN KEY ([OrderDate])
REFERENCES [Rasz].[Dim_Time]([Time])

ALTER TABLE [Rasz].[Fact_Sales]
ADD CONSTRAINT FK_FactSales_ShipDate_DimTime_Time FOREIGN KEY ([ShipDate])
REFERENCES [Rasz].[Dim_Time]([Time])
```

## Dodawanie danych:

```
with Dates(Date) as (
	select OrderDate
	from Sales.SalesOrderHeader
	union
	select ShipDate
	from Sales.SalesOrderHeader
),
Months(MonthNum, MonthName) as (
	select *
	from (values (1, 'January'),
		(2, 'February'),
		(3, 'March'),
		(4, 'April'),
		(5, 'May'),
		(6, 'June'),
		(7, 'July'),
		(8, 'August'),
		(9, 'September'),
		(10, 'October'),
		(11, 'November'),
		(12, 'December')) AS M(C1, C2)
),
Weekdays(DayNum, Weekday) as (
	select *
	from (values (2, 'Monday'),
		(3, 'Tuesday'),
		(4, 'Wednesday'),
		(5, 'Thursday'),
		(6, 'Friday'),
		(7, 'Saturday'),
		(1, 'Sunday')) as W(C1, C2)
)
INSERT INTO [Rasz].[Dim_Time](Time, Year, Quarter, Month, MonthWord, Weekday,
				DayInMonth)
(
	select
		YEAR(Date) * 10000 +
			MONTH(Date) * 100 +
				DAY(Date),
		YEAR(Date),
		MONTH(Date) / 4 + 1,
		MONTH(Date),
		MonthName,
		Weekday,
		DAY(Date)
	from Dates D
	join Months M
		on MONTH(D.Date) = M.MonthNum
	join Weekdays W
		on DATEPART(WEEKDAY, D.Date) = W.DayNum
);
```

Do wprowadzenia nazw miesięcy oraz dni zostały użyte dodatkowe tabele dołączane do zapytania. Łatwiejszym sposobem byłoby użycie funkcji DATENAME() oraz FORMAT().

## Zad. 3. Elementarne czyszczenie danych

Zamienić wszystkie wartości NULL:

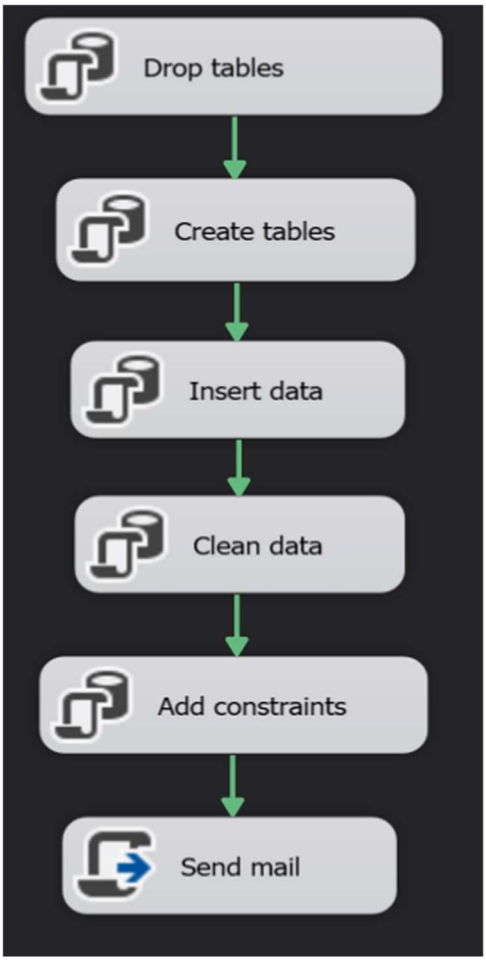
- w kolumnie Color (tabela DIM_PRODUCT) na „Unknown”,
- w kolumnie SubCategoryName (tabela DIM_PRODUCT) na „Unknown”.
- w kolumnie CountryRegionCode na 000,
- w kolumnie Group na „Unknown”

```
UPDATE [Rasz].[Dim_Product]
SET Color = 'Unknown'
WHERE Color is NULL;

UPDATE [Rasz].[Dim_Product]
SET SubCategoryName = 'Unknown'
WHERE SubCategoryName is NULL;

UPDATE [Rasz].[Dim_Customer]
SET CountryRegionCode = '000'
where CountryRegionCode is NULL;

UPDATE [Rasz].[Dim_Salesperson]
SET CountryRegionCode = '000'
where CountryRegionCode is NULL;

UPDATE [Rasz].[Dim_Customer]
SET [Group] = 'Unknown'
where [Group] is NULL;

UPDATE [Rasz].[Dim_Salesperson]
SET [Group] = 'Unknown'
where [Group] is NULL;
```

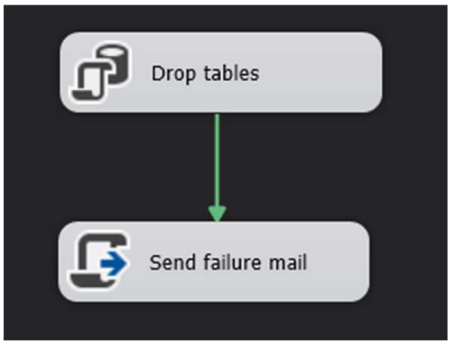
# Zad. 4. Proces ETL

Używając Visual Studio utworzyć projekt typu Integration Services.

Control Flow:




Event handlers:




## Drop tables

Krok Drop tables usuwa wszystkie istniejące tabele Dim oraz Fact, jak opisano w zadaniu 1. W przypadku obsługi zdarzeń krok ten jest identyczny.

## Create tables

W kroku tym tworzone są wszystkie tabele, bez kluczy obcych oraz bez wprowadzania danych.

## Insert data

Za pomocą skryptów opisanych w poprzednim ćwiczeniu oraz tworzenia danych dla tabeli Dim_time wprowadzane są dane (jeszcze nie wyczyszczone) do utworzonych tabel.

## Clean data

W kroku tym usuwane są duplikaty adresów w tabeli Dim_Customer (opisane w poprzedniej liście) oraz zamieniane są wartości NULL na dane napisowe jak w zadaniu 3.

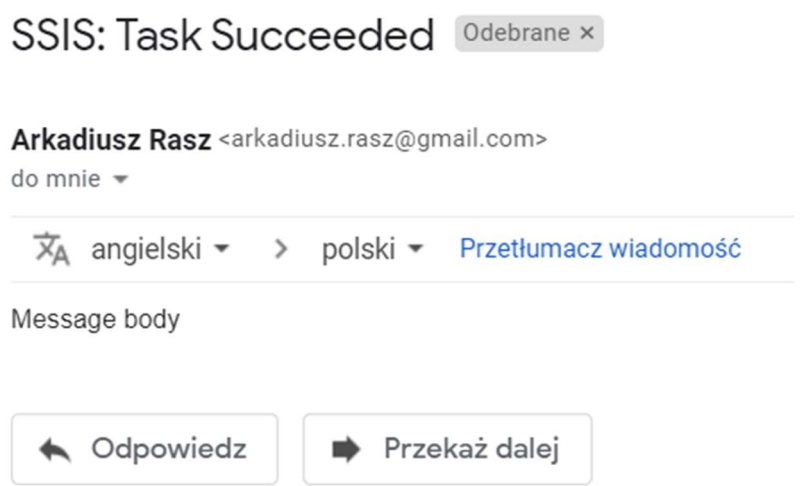
## Add constraints

Dodawane są wszystkie klucze obce do tabeli faktów.

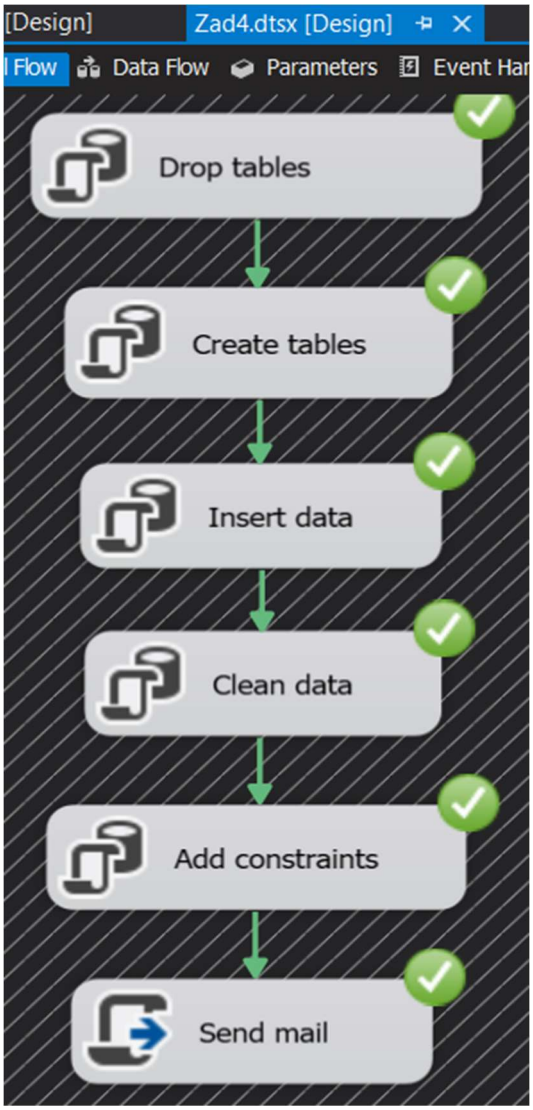
## Send Mail

```
var from = new MailAddress("arkadiusz.rasz@gmail.com", "Arkadiusz Rasz");
var to = new MailAddress("arkadiusz.rasz@gmail.com");
var password = "**********";
var smtp = new SmtpClient
{
   Host = "smtp.gmail.com",
   Port = 587,
   EnableSsl = true,
   DeliveryMethod = SmtpDeliveryMethod.Network,
   UseDefaultCredentials = false,
   Credentials = new NetworkCredential(from.Address, password),
   Timeout = 20000
};

var msg = new MailMessage
{
   IsBodyHtml = true,
   Subject = "SSIS: Task Succeeded",
   Body = "Message body",
   From = from
};
msg.To.Add(to);
{
   smtp.Send(msg);
}

TaskResult = (int)ScriptResults.Success;
```

Z powodu niepewności czy poczta studencka nie odfiltruje maila, użyto personalnego maila na poczcie gmail. Przykład wiadomości po ukończeniu procesu:



Po uruchomieniu pakietu, proces wykonuje się poprawnie:




Tabele w bazie danych są wypełnione:

```
25
26  select * from Rasz.Fact_Sales
```

108 %

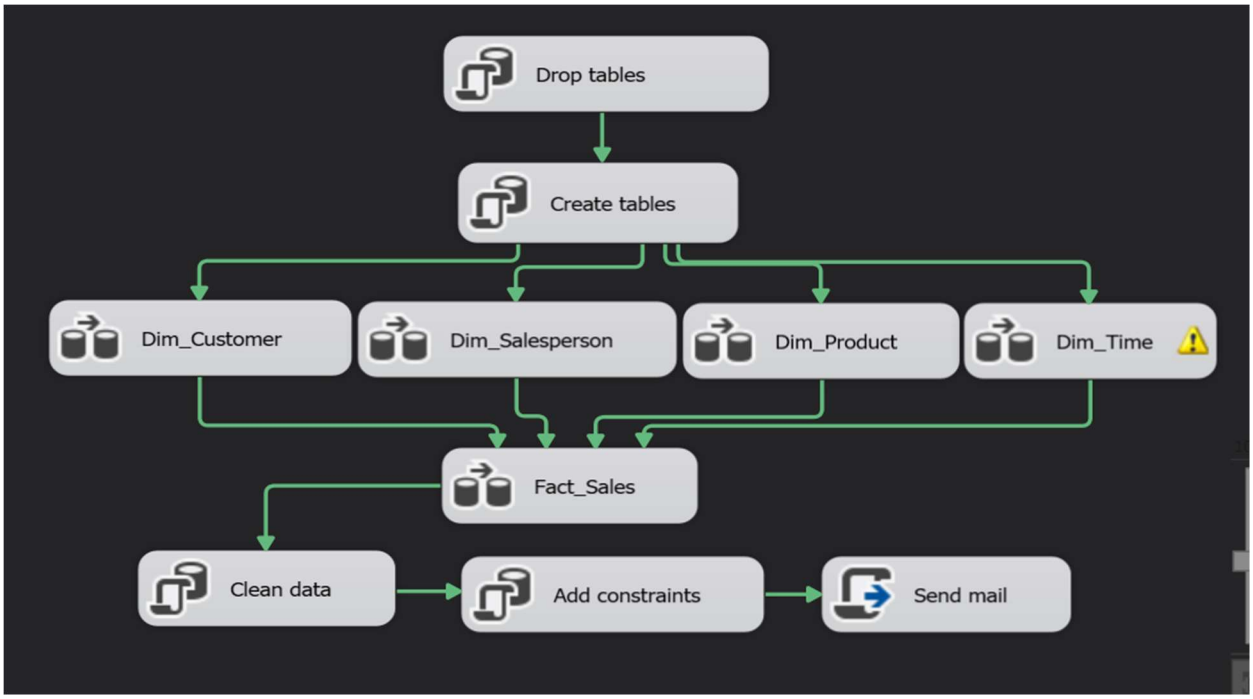
Results | Messages

| | ProductID | CustomerID | SalesPersonID | OrderDate | ShipDate | OrderQty | UnitPrice | UnitPriceDiscount | LineTotal |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 776 | 29825 | 279 | 20110531 | 20110607 | 1 | 2024.994 | 0.00 | 2024.994000 |
| 2 | 777 | 29825 | 279 | 20110531 | 20110607 | 3 | 2024.994 | 0.00 | 6074.982000 |
| 3 | 778 | 29825 | 279 | 20110531 | 20110607 | 1 | 2024.994 | 0.00 | 2024.994000 |
| 4 | 771 | 29825 | 279 | 20110531 | 20110607 | 1 | 2039.994 | 0.00 | 2039.994000 |
| 5 | 772 | 29825 | 279 | 20110531 | 20110607 | 1 | 2039.994 | 0.00 | 2039.994000 |
| 6 | 773 | 29825 | 279 | 20110531 | 20110607 | 2 | 2039.994 | 0.00 | 4079.988000 |
| 7 | 774 | 29825 | 279 | 20110531 | 20110607 | 1 | 2039.994 | 0.00 | 2039.994000 |
| 8 | 714 | 29825 | 279 | 20110531 | 20110607 | 3 | 28.8404 | 0.00 | 86.521200 |
| 9 | 716 | 29825 | 279 | 20110531 | 20110607 | 1 | 28.8404 | 0.00 | 28.840400 |
| 10 | 709 | 29825 | 279 | 20110531 | 20110607 | 6 | 5.70 | 0.00 | 34.200000 |
| 11 | 712 | 29825 | 279 | 20110531 | 20110607 | 2 | 5.1865 | 0.00 | 10.373000 |
| 12 | 711 | 29825 | 279 | 20110531 | 20110607 | 4 | 20.1865 | 0.00 | 80.746000 |
| 13 | 762 | 29672 | 279 | 20110531 | 20110607 | 1 | 419.4589 | 0.00 | 419.458900 |

# Zad. 6. ETL bez SQLa

## Schemat ETL

W ramach zadania, zapytania SQL służące do wypełniania utworzonych tabel danymi zastąpione są zadaniem „Data Flow”, pozwalającym na tworzenie łączeń i operacji na przepływie danych bez używania języka zapytań. Schemat całego procesu ETL jest podobny jak wcześniej, lecz zadanie Insert data zostało zastąpione pojedynczymi zadaniami Data Flow odpowiednio dla każdej tabeli.

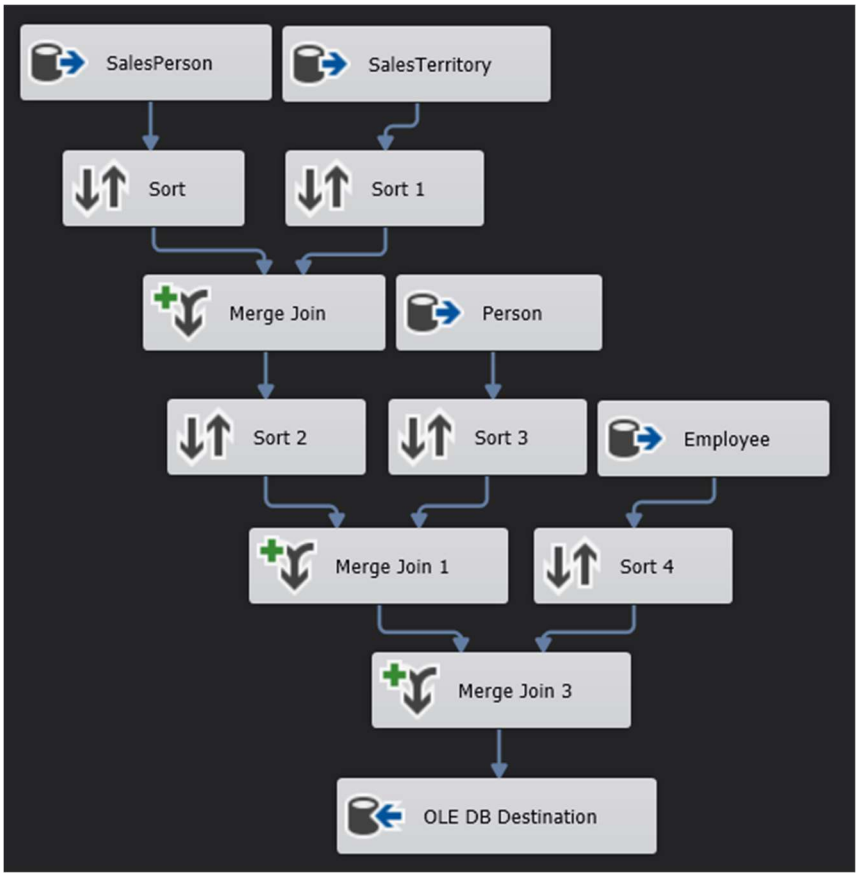



Poniżej przedstawiony jest przepływ danych w każdym utworzonym elemencie Data Flow

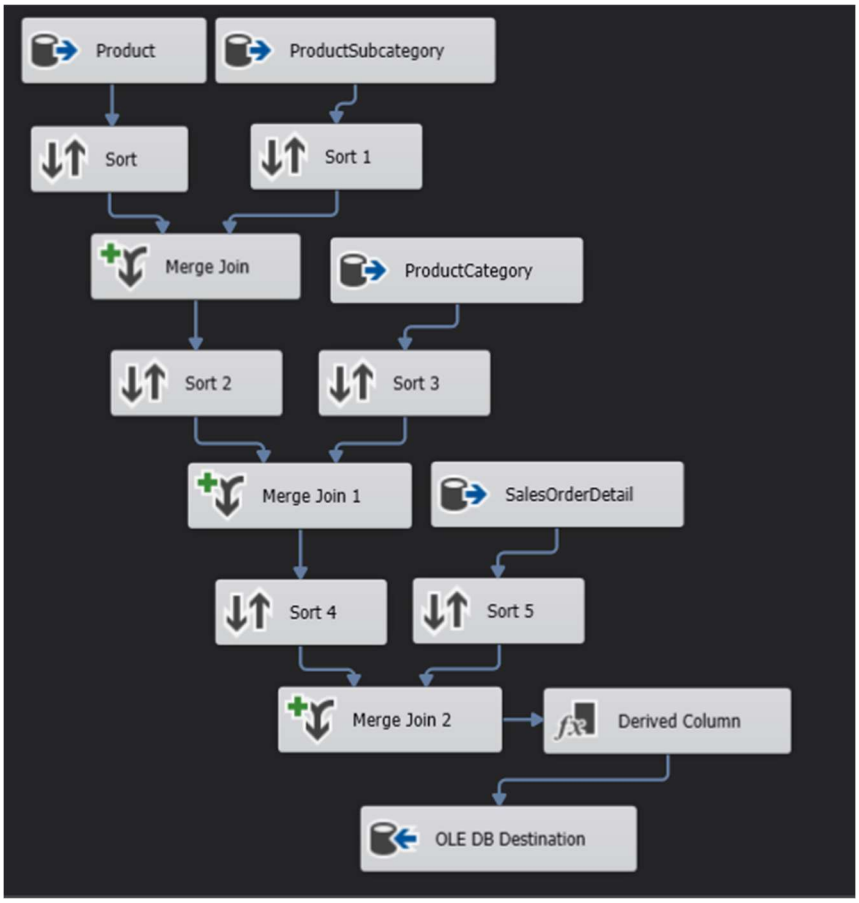
## Elementy Data Flow

### Dim_Customer:

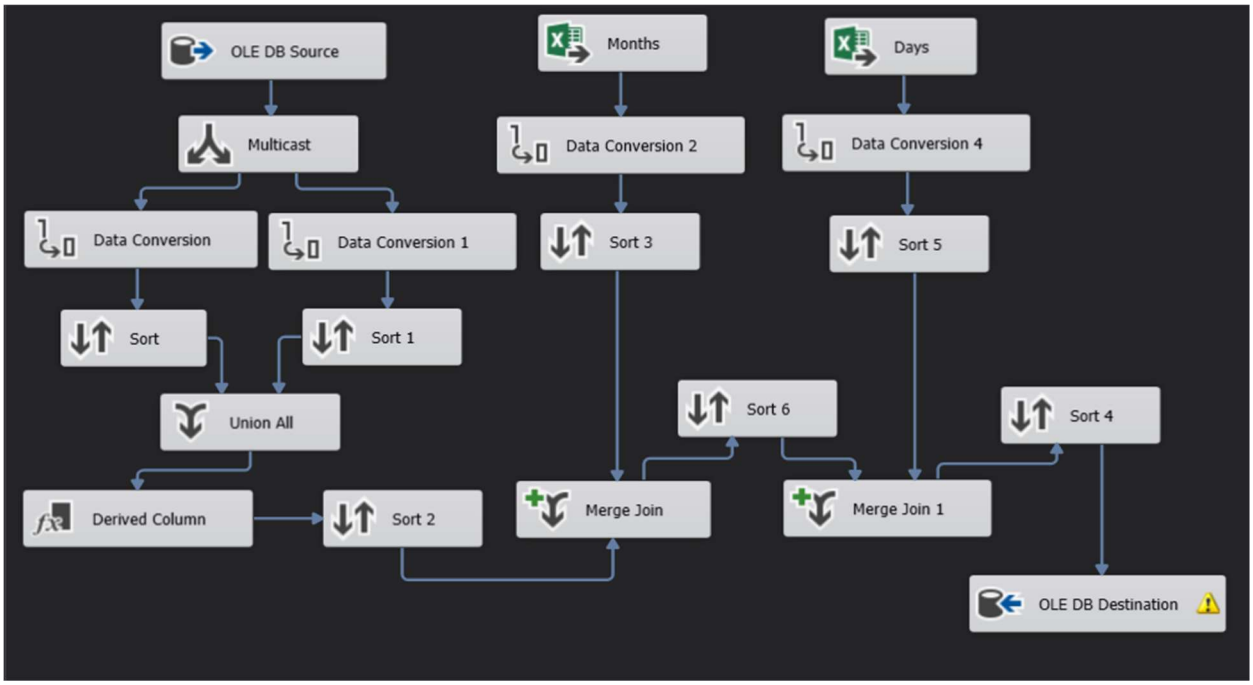
## Dim_Salesperson



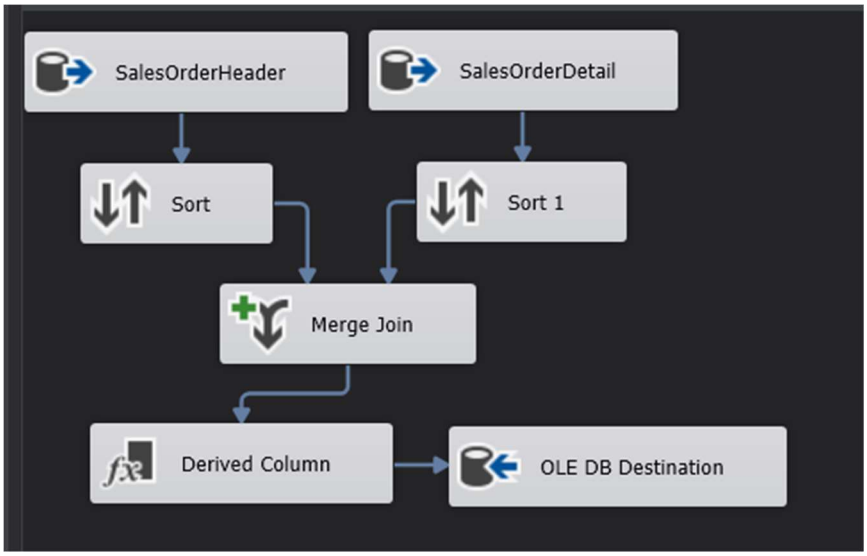

## Dim_Product



## Dim_Time




## Fact_Sales




Niestety, procesu nie udało się uruchomić i sprawdzić poprawności wstawiania danych. Visual Studio zamykał się za przy każdej próbie uruchomienia paczki. Dim_Customer oraz Dim_Salesperson zostały przetestowane pojedynczo i wprowadzają dane takie same, jak poprzednie zapytania SQL. Program nie potrafił już poradzić sobie z Dim_Customer.

Jest to prawdopodobnie spowodowane dużą ilością danych w przepływie w połączeniu z ogarniczeniem maszyny wirtualnej, na której działa system. Ma ona dostępne 6 GB RAM, lecz przy samym działającym Visual Studio, pamięć jest w większości używana.

## Podsumowanie i wnioski

Proces ETL pozwala na automatyzację procesu tworzenia tabel wymiarów i faktów oraz denormalizacji danych. Jest szybkie, a w przypadku nie zmieniającej się struktury bazy danych, pozwala zapomnieć o całym procesie oraz skupić się na prawdziwej analizy danych na utworzonej już infrastrukturze. Najprawdopodobniej za pomocą skryptów jest również możliwość dalszej automatyzacji poprzez automatyczne tworzenie kostki zdefiniowanej w osobnym projekcie.

Narzędzia Data Flow dostępne w Integration Services pozwalają na uniknięcie języka zapytań i „wyklikiwania" całego przepływu danych. Może być zachęcające dla wielu osób, lecz moim zdaniem proces ten jest bardziej problematyczny niż pisanie własnych zapytań SQL. Brakuje prostych opcji, które znacznie przyspieszyłyby proces, jak możliwość sortowania wyjścia danego elementu, zamiast tworzenia nowego elementu sortowania w każdym przypadku, kiedy potrzeba złączyć dane. Pojawiające się często informacje o błędach nie są przydatne, a najczęstszym rozwiązaniem problemów jest usunięcie i ponowne dodanie elementu stwarzającego problem. Cały przepływ danych jest też bardziej zasobochłonny, przez co proces trwa dłużej, a nawet wyłącza program Visual Studio.