/\*

Objaśnienia:

Przedstawione rozwiązania dotyczą wersji MS SQL 2016(lub nowszej). Możliwość skorzystania z OPENROWSET wymaga zainstalowania sterowników ODBC lub posiadania odpowiednich bibliotek OLE DB. W podanym rozwiązaniu dostawcą OLE DB jest Microsoft.Jet.OLEDB.12.0. Wybranie odpowiedniego dostawcy wymaga sprawdzenia w bazie jacy dostawcy są zainstalowani na serwerze. Można ich odnaleźć (ręcznie) w drzewie hierarchii obiektów klikając kolejno Databases-Server Objects-Linked Servers-Providers. Można też ich wyświetlić za pomocą poniższego polecenia : EXEC xp\_enum\_oledb\_providers. Trzeba też podać wersję excela w odpowiednim formacie np. Excel 2007 to wersja Excela 12.0. Informacje te można znaleźć na stronach dokumentacji technicznej Microsoft.

Przyjęto, że plik przesłany przez firmę zewnętrzną zapisywany jest na pulpicie.

Zapytania zostały poprzedzone wskazaniem na wykorzystanie bazy Magazynowanie. Tak by mogły być wykonane niezależnie od siebie a niekoniecznie w jednym ciągu następujących po sobie operacji.

W rozwiązaniu pominięto tworzenie dedykowanego schematu w bazie danych. Schemat ułatwia administrowanie dostępem do poszczególnych obiektów w bazie. Z racji tego, że rozwiązanie ma charakter demonstracyjny i nie występują żadni stali użytkownicy w bazie nie utworzono dedykowanego schematu. Wszystkie obiekty zastaną utworzone w domyślnym schemacie dbo, który nawet jeśli nie zostanie podany to SQL Server przypisze go jako schemat domyślny. Zakładamy też, że osoby mogące korzystać z przedstawionych rozwiązań nie należą do grupy w systemie operacyjnym Windows.

(Działa dla SQL 2016 i 2014) Do sprawdzenia czy istnieje dany obiekt w bazie została wykorzystana konstrukcja If OBJECT\_ID. Sprawdza ona czy istnieje obiekt o dokładnie takiej samej nazwie jeśli tak to likwiduje go i następnie wykonuje się polecenie utworzenia danego obiektu. Wybranie tej konstrukcji wynika z założenia, iż jeżeli istnieje obiekt o tej nazwie to został utworzony w wyniku błędu. Stąd konieczność zniszczenia go i utworzenia prawidłowego.

Zakładamy, że całość działań jest wykonywana z komputera pracownika Jana Kowalskiego, który ma uprawnienia pozwalające na wykonanie poniższego kodu. Jego numer identyfikacyjny w firmie składa się z inicjałów imienia i nazwiska oraz numeru pracownika co razem daje JK43. Jego numer identyfikacyjny jest jednocześnie jego loginem do systemu Windows

\*/

IF DB\_ID(' Magazynowanie ') IS NOT NULL DROP DATABASE Magazynowanie;

--sprawdza czy istnieje baza danych Magazynowanie jeśli tak to likwiduje ją--

CREATE DATABASE Magazynowanie

GO

IF OBJECT\_ID('dbo.Taryfa') is not null DROP TABLE dbo.Taryfa;

-- sprawdza czy istnieje tabela taryfa jesli tak to liwkduje ją—-

USE Magazynowanie;

GO

CREATE TABLE Taryfa (ID int IDENTITY(1,1) PRIMARY KEY, Stawka decimal(8,7) not null UNIQUE);

--Tworzenie tabeli ze stawkami magazynowymi--

INSERT INTO Magazynowanie.dbo.Taryfa (Stawka) VALUES (0.00060), (0.0007), (0.0015),(0.0085),(0.0004),(0.001),(0.0003),(0.0005),(0.0019),(0.0002);

--Wstawia stawki do tabeli ze stawkami magazynowymi—-

SELECT\*FROM Magazynowanie.dbo.Taryfa;

--sprawdza co zostało wstawione do tabeli taryfa--

IF OBJECT\_ID('dbo.Magazyn') is not null DROP TABLE dbo.magazyn;

-- sprawdza czy istnieje tabela magazyn jeśli tak to likwiduje ją--

USE Magazynowanie;

GO

CREATE TABLE dbo.magazyn (ID int IDENTITY(1,1), kodTowaru char(500) , nazwaTowaru varchar(500) , klient nvarchar(500) , KAM varchar(100) , IDtaryfaMag integer ,

JM char(3) , iloscWMAG float , DataWaznosci date , Trwaloscproduktu int , CONSTRAINT FK\_Taryfa FOREIGN KEY (IDtaryfaMag) REFERENCES Magazynowanie.dbo.Tafya (ID) PRIMARY KEY(kodTowaru, nazwaTowaru, klient, KAM, IDtaryfaMag, JM, iloscWMAG, DataWaznosci, Trwaloscproduktu,));

--Tworzenie tabeli w formacie danych, który zrzuca program ERP lub dowolny magazynowy —-

/\*

klucz główny został utworzony na prawie wszystkich kolumnach tabeli pełni on funkcję chroniącą przed wrzuceniem duplikatów do tabeli

\*/

IF OBJECT\_ID('uspPobierzDane ') is not null DROP PROC dbo.uspPobierzDane;

USE Magazynowanie;

GO

CREATE PROC dbo.uspPobierzDane

AS BEGIN

EXEC sp\_configure 'show advanced options', 1;

RECONFIGURE WITH OVERRIDE;

EXEC sp\_configure 'Ad Hoc Distributed Queries', 1;

RECONFIGURE WITH OVERRIDE;

--uruchamia możliwość skorzystania z OPENROWSET i wyciągnięcia danych z dowolnego pliku--

INSERT INTO Magazynowanie.dbo.Magazyn (kodTowaru, nazwaTowaru, klient, KAM, IDtaryfaMag,

JM, iloscWMAG, DataWaznosci, Trwaloscproduktu)

SELECT \* FROM OPENROWSET('Microsoft.ACE.OLEDB.12.0',

'Excel 12.0; Database=C:\Users\JK43\Desktop\Magazyn.xlsx', 'SELECT \* FROM [Magazyn$]')

END;

/\*

importuje dane z pliku zrzuconego przez ERP lub program magazynowy do tabeli magazyn przy założeniu, że zrzut z ERP arkusz z danymi nazywa Magazyn i zrzuca dane na pulpit

\*/

SELECT\*FROM Magazynowanie.dbo.Magazyn;

--sprawdza co zostało wrzucone do tabeli--

IF OBJECT\_ID('vRapMagazyn') is not null DROP VIEW dbo.vRapMagazyn;

--sprawdza czy istnieje widok Rapmagazyn jeśli tak likwiduje go--

USE Magazynowanie;

GO

CREATE VIEW dbo.vRapMagazyn

AS

SELECT kodTowaru, nazwaTowaru,klient, KAM, IDtaryfaMag, JM, iloscWMAG, DataWaznosci, Trwaloscproduktu,dateadd(day,-Trwaloscproduktu,DataWaznosci) AS ProdDate, GetDate() AS dzis, datediff(day,dateadd(day,-Trwaloscproduktu,DataWaznosci), GetDate()) AS MagazDni,

iloscWMAG\*(datediff(day,dateadd(day,-Trwaloscproduktu,DataWaznosci),GetDate()))\*stawka AS oplataMagazynowa

FROM Magazynowanie.dbo.magazyn INNER JOIN Magazynowanie.dbo.Taryfa AS c ON IDtaryfaMag=c.ID;

--Tworzenie widoku, który przechowuje wszystkie potrzebne do raportowania dane--

IF OBJECT\_ID('uspKosztyMagNaKlienta') is not null DROP PROC dbo.uspKosztyMagNaKlienta;

--sprawdza czy taka procedura już nie istnieje jeśli tak to ją likwiduje--

USE Magazynowanie;

GO

CREATE PROC dbo.uspKosztyMagNaKlienta (@nazwaklienta nvarchar (500))

AS

SELECT klient, sum(oplataMagazynowa) AS KosztMagazynowania

FROM Magazynowanie.dbo.vRapMagazyn WHERE klient=@nazwaklienta GROUP BY klient;

/\*

Tworzy procedurę składową, która dla podanej nazwy klienta zwraca jego całkowity koszt magazynowania, może być wykorzystane przez aplikację webową do wyświetlania na stronie

\*/

IF OBJECT\_ID('uspCzyscMagazyn') is not null DROP PROC dbo.uspCzyscMagazyn;

GO

USE Magazynowanie;

GO

CREATE PROC dbo.uspCzyscMagazyn

AS

BEGIN

DELETE FROM Magazynowanie.dbo.Magazyn

END;

/\*

Czyści tabelę magazyn przed każdym nowy importem danych trzeba ją uruchomić i usunąć poprzednie wpisy. Zakładamy, że raport zewnętrzny będący źródłem danych zawsze przedstawia aktualny stan magazynowy będący podstawą do raportowania

\*/