

**Politechnika Świętokrzyska**

**Informatyka, studia stacjonarne, semestr VII**

**Nowoczesne systemy przetwarzania danych – Projekt**

**Bartłomiej Kotarski, Marcin Krocak**

**Grupa 4ID11A**

## Spis treści

1.	WSTĘP .....	3
2.	SCHEMAT BAZY DANYCH .....	6
3.	GENERATOR DANYCH LOSOWYCH .....	8
4.	IMPORT DANYCH LOSOWYCH DO BAZY .....	9
5.	KOSTKA ANALITYCZNA .....	10
6.	RAPORTY.....	11
7.	WNIOSKI.....	14

# 1. Wstęp

Tematem naszego projektu jest baza danych dla zajezdni autobusowej. Firma ta zajmuje się zorganizowanym przewozem osób po określonych trasach. W firmie zatrudnieni są pracownicy, nie są to wyłącznie kierowcy. W związku z tym, że pracownicy objęci są pewną umową, konieczne jest posiadanie informacji na ich temat. Informacjami takimi są między innymi dane ich zamieszkania, a także informacje zawarte w dowodach osobistych. Dane o dowodach pozyskiwane będą wyłącznie od pracowników, którzy są kierowcami. Trasy jakimi przemieszczają się autobusy są ściśle określone, istotne są zatem informacje na temat miejsca odjazdu i przyjazdu, a także konkretna cena. Cena może być różna w zależności od tego czy bilet jest normalny czy ulgowy. Flota pojazdów jaką dysponuje firma też musi być określona. Istotnymi danymi o autobusach jest między innymi liczba miejsc. Są nimi, poza tym parametry pojazdu takie jak moc silnika, marka i model. Firma może posiadać pojazdy wymagające nie tylko jednej kategorii, istotna jest też wiedza o tym jaką kategorię ma każdy z zatrudnionych kierowców. Kluczowym wydarzeniem będą kursy autobusów. Ważna jest wiedza na temat tego kto prowadził, którym pojazdem i kiedy. W związku z tym, że firma prowadzi działalność dla zarobku potrzebne są też informacje na temat zysku z biletów normalnych i ulgowych. Do planowania tego jaki rodzaj autobusu wystawić o której godzinie użyteczna może być też informacja o tym jakie było wypełnienie pojazdu przez pasażerów.

Założenia te prowadzą do pierwszego problemu. Należy zaprojektować stosowną hurtownię, określić typ jej struktury, tabele, relacje między nimi i dane w nich zawarte. Zdecydowaliśmy się na schemat płatka śniegu. Schemat gwiazdy mógłby okazać się za mały, a konstelacyjny zbyt złożony na skalę problemu. Należy wyodrębnić tabelę faktów i wymiarów. Wydarzeniem związanym z tego typu firmą jest kurs, w związku z tym faktem ona zostanie tabelą faktów. Dane o czasie, trasach, pojazdach i pracownikach zamieszczone będą w tabelach wymiarów. Wypełnia to wszystkie wymagania związane z założeniami i tymi które mają spełniać tabele wymiarów – co, gdzie i kiedy. Tabele zostały wypełnione zgodnymi z założeniami kolumnami i połączone relacjami jeden do wielu. Ustalone zostały klucze główne. Selecty w języku SQL przedstawione są poniżej. Zdecydowaliśmy się na taki sposób stworzenia bazy, ponieważ daje możliwość ponownego użycia w razie problemów z bazą. W przypadku ręcznego tworzenia tabel byłaby konieczność ponownego „przeklikania” programu celem wykonania kluczy obcych.

```

create table dowody_osobiste(
    id int not null primary key,
    imie varchar(40) not null,
    nazwisko varchar(40) not null,
    pesel varchar(40) not null,
    numer_dowodu varchar(40) not null,
);

create table kierowcy (
    id int not null primary key,
    prawo_jazdy varchar(40) not null,
    id_dowodu int not null references dowody_osobiste(id),
);

create table adresy(
    id int not null primary key,
    miasto varchar(40) not null,
    ulica varchar(40) not null,
    numer_domu varchar(40) not null,
    kod_pocztowy varchar(40) not null,
);

create table pracownicy(
    id int not null primary key,
    id_kierowcy int not null references kierowcy(id),
    id_adresu int not null references adresy(id),
    pensja_netto float not null,
    rok_podpisania_umowy int not null,
    miesiac_podpisania_umowy int not null,
    dzien_podpisania_umowy int not null,
);

create table trasy(
    id int not null primary key,
    miasto_odjazdu varchar(100) not null,
    miasto_przyjazdu varchar(100) not null,
    cena_biletu_nor int not null,
    cena_biletu_ulg int not null,
);

create table czas(
    id int not null primary key,
    godzina_odjazdu int not null,
    minuty_odjazdu int not null,
    godzina_przyjazdu int not null,
    minuty_przyjazdu int not null,
    rok_kursu int not null,
    miesiac_kursu int not null,
    dzien_kursu int not null,
);

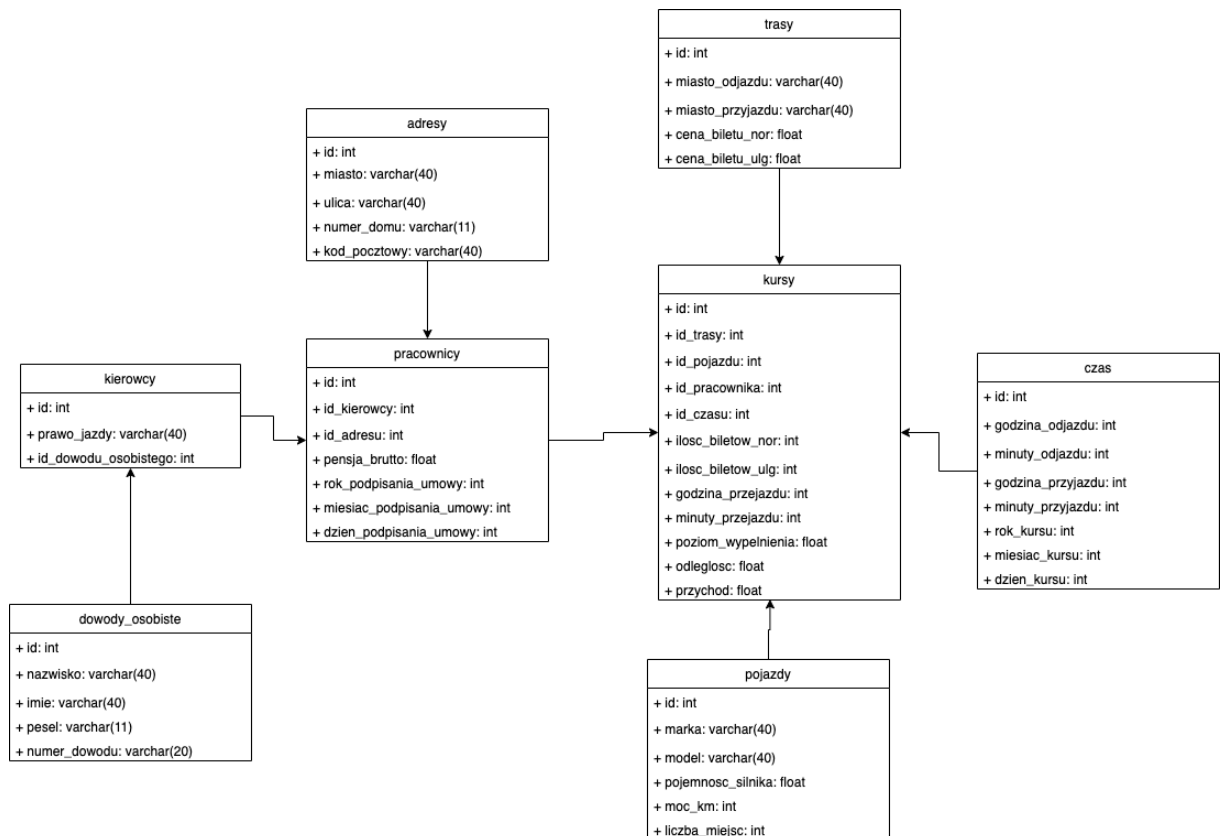
create table pojazdy(
    id int not null primary key,
    marka varchar(40) not null,

```

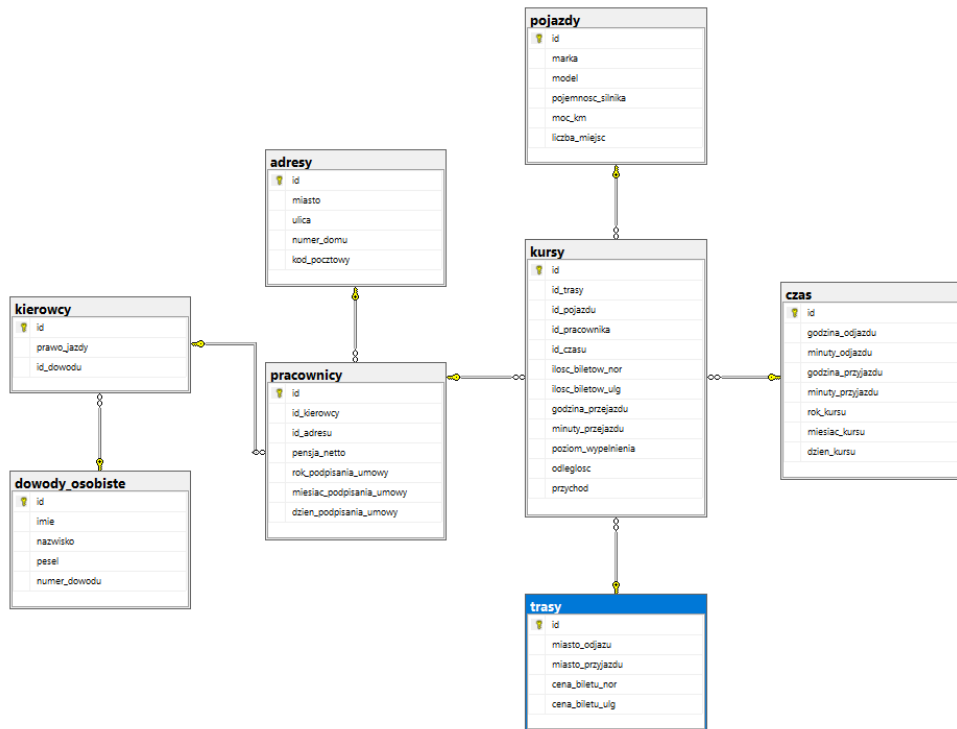
```
model varchar(40) not null,  
pojemnosc_silnika float not null,  
moc_km int not null,  
liczba_miejsc int not null,  
);  
  
create table kursy (  
    id int not null primary key,  
    id_trasy int references trasy(id),  
    id_pojazdu int references pojazdy(id),  
    id_pracownika int references pracownicy(id),  
    id_czasu int references czas(id),  
    ilosc_biletow_nor int not null,  
    ilosc_biletow_ulg int not null,  
    godzina_przejazdu int not null,  
    minuty_przejazdu int not null,  
    poziom_wypelnienia float not null,  
    odleglosc float not null,  
    przychod float not null,  
);
```

## 2. Schemat bazy danych

Na początku pracy przed stworzeniem tabel należało zaprojektować odpowiednią strukturę bazy danych. Został on zaprojektowany z użyciem przeglądarkowego narzędzia Diagram.io i był pierwszym spojrzeniem na projekt.

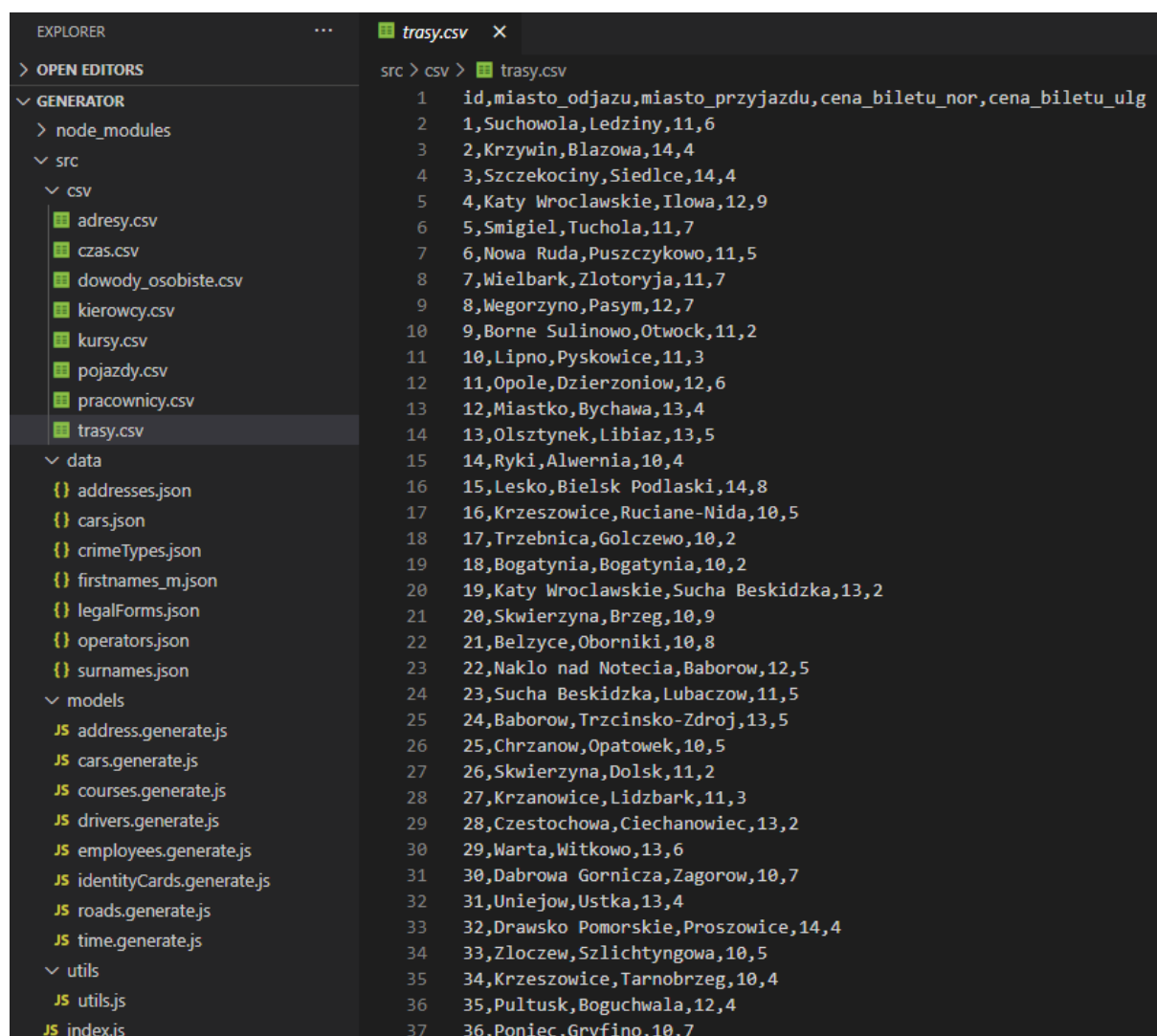


Następnie w związku z utworzeniem tabel można przejść do wygenerowania diagramu bazy. Jest on generowany przez MSSQL Management ponieważ daje on natychmiastowy rezultat. Po utworzeniu diagram jest zgodny z tym co wywołaliśmy kodem w SQL.



### 3. Generator danych losowych

Następnym problemem jest utworzenie danych w formacie .csv które zostaną następnie załadowane do bazy. Generator utworzony został przy użyciu języka JavaScript w środowisku uruchomieniowym Node.js. Wybraliśmy taką metodę, ponieważ liczba danych jest bardzo duża, wynosząca tysiące rekordów. Tworzenie tego ręcznie lub za pomocą arkusza kalkulacyjnego byłoby możliwe, lecz czasochłonne. Zaimplementowanie tego w sposób właściwy eliminuje też problemy z powiązaniem kluczy obcych.



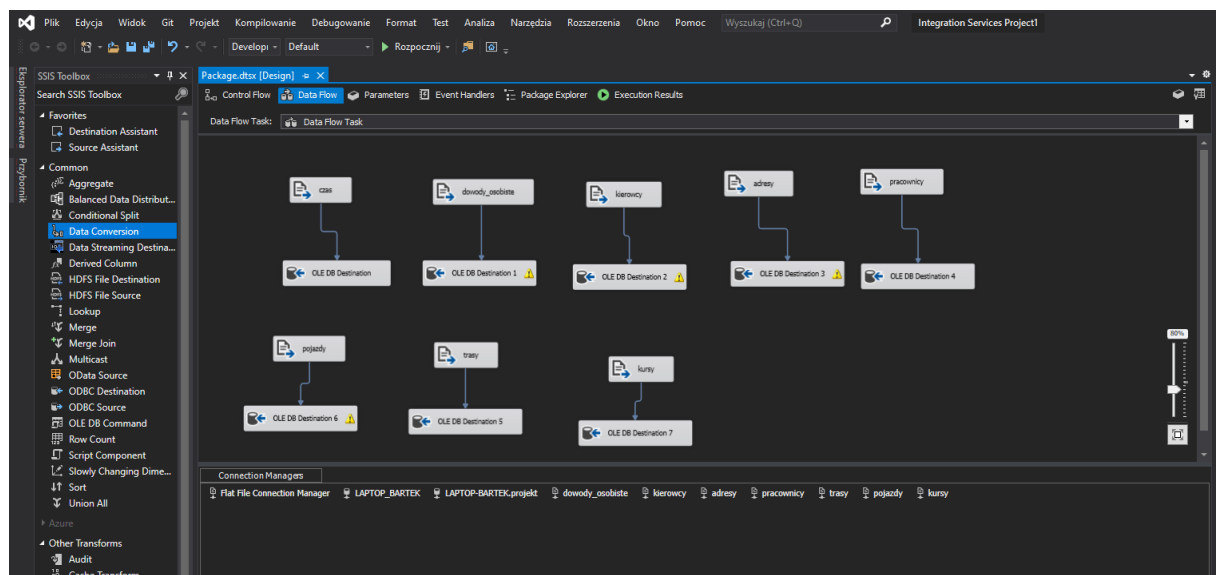
```
EXPLORER
> OPEN EDITORS
  GENERATOR
  > node_modules
  > src
    > csv
      adresy.csv
      czas.csv
      dowody_osobiste.csv
      kierowcy.csv
      kursy.csv
      pojazdy.csv
      pracownicy.csv
      trasy.csv
  > data
    {} addresses.json
    {} cars.json
    {} crimeTypes.json
    {} firstnames_m.json
    {} legalForms.json
    {} operators.json
    {} surnames.json
  > models
    JS address.generate.js
    JS cars.generate.js
    JS courses.generate.js
    JS drivers.generate.js
    JS employees.generate.js
    JS identityCards.generate.js
    JS roads.generate.js
    JS time.generate.js
  > utils
    JS utils.js
  JS index.js

src > csv > trasy.csv
1 id,miasto_odjazdu,miasto_przyjazdu,cena_biletu_nor,cena_biletu_ulg
2 1,Suchowola,Ledziny,11,6
3 2,Krzywin,Blazowa,14,4
4 3,Szczekociny,Siedlce,14,4
5 4,Katy Wroclawskie,Ilowa,12,9
6 5,Smigiel,Tuchola,11,7
7 6,Nowa Ruda,Puszczykowo,11,5
8 7,Wielbark,Zlotoryja,11,7
9 8,Wegorzyno,Pasym,12,7
10 9,Borne Sulinowo,Otwock,11,2
11 10,Lipno,Pyskowice,11,3
12 11,Opole,Dzierzoniow,12,6
13 12,Miastko,Bychawa,13,4
14 13,Olsztynek,Libiaz,13,5
15 14,Ryki,Alwernia,10,4
16 15,Lesko,Bielsk Podlaski,14,8
17 16,Krzeszowice,Ruciane-Nida,10,5
18 17,Trzebnica,Golczewo,10,2
19 18,Bogatynia,Bogatynia,10,2
20 19,Katy Wroclawskie,Sucha Beskidzka,13,2
21 20,Skwierzyna,Brzeg,10,9
22 21,Belzyce,Oborniki,10,8
23 22,Naklo nad Notecia,Baborow,12,5
24 23,Sucha Beskidzka,Lubaczow,11,5
25 24,Baborow,Trzcinsko-Zdroj,13,5
26 25,Chrzanow,Opatowek,10,5
27 26,Skwierzyna,Dolsk,11,2
28 27,Krzanowice,Lidzbark,11,3
29 28,Czestochowa,Ciechanowiec,13,2
30 29,Warta,Witkowo,13,6
31 30,Dabrowa Gornicza,Zagorow,10,7
32 31,Uniejow,Ustka,13,4
33 32,Drawsko Pomorskie,Proszowice,14,4
34 33,Zloczew,Szlichtyngowa,10,5
35 34,Krzeszowice,Tarnobrzeg,10,4
36 35,Pultusk,Boguchwala,12,4
37 36,Poniec,Gryfino,10,7
```



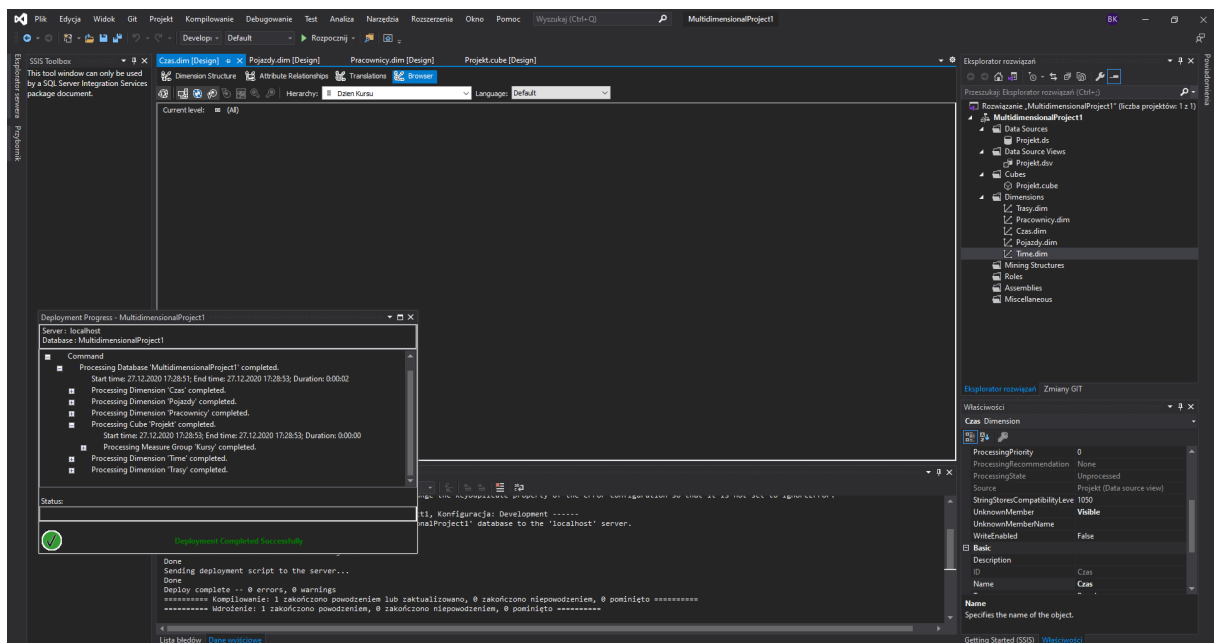
## 4. Import danych losowych do bazy

Po wygenerowaniu danych przyszedł czas na umieszczenie ich w hurtowni. Do tego zadania można było podejść na kilka sposobów, między innymi poprzez narzędzia dostępne w MSSQL Server Management Studio. Użyte zostało jednak IDE Visual Studio i narzędzia dostępne z dzięki SSIS. Jest to lepsze rozwiązanie, ponieważ narzędzie pozwala zautomatyzować proces i uniknąć powstawania możliwych błędów. Gdy połączenie zostało nawiązane pomyślnie uruchomienie programu oznaczało zakończenie zadania.



## 5. Kostka analityczna

Następnym zadaniem było wykonanie kostki analitycznej będącej później punktem wejściowym do generowania raportów. Zadanie zrealizowane zostało przy użyciu usługi SSAS w Visual Studio. Prostota i ilość dostępnych narzędzi przesądziła o takim wyborze. Dzięki narzędziom do wielowymiarowej analizy danych prezentowanym przez kostkę CUBE można było ostatecznie tworzyć raporty. Po stworzeniu projektu i pomyślnym połączeniu się ze źródłem danych przyszedł czas na wykonanie kostki. Została stworzona z istniejących tabel, wybrana została tabela faktów jaką są kursy. Z myślą o przyszłych raportach wybrane zostały miary i tabele wymiarów. Wykonanie tabeli czasu dla kostki było również niezbędne do sporządzenia raportów biorących za wyznacznik czas. Problematicznym etapem było debugowanie kodu, aby móc wgrać ją na serwer. Po ustaleniu przyczyn jaką były kwestie uprawnień użytkownika debugowanie przebiegło bez problemów.



## 6. Raporty

Ostatnim etapem projektu było wykonanie raportów. Stanowią one cel generowania kostki analitycznej. Do ich wykonania niezbędne były usługi związane z SSRS w Visual Studio. Po utworzeniu projektu należało określić źródło danych jakim jest kostka. Wybrano serwer lokalny oraz wybrano źródło Microsoft SQL Server Analysis Services. Pomyślny test połączenia i wybór autoryzacji oznaczał prawidłowe połączenie z kostką. Stworzone zostały raporty opierające się na Zestawach danych. Są nimi zapytania zawierające określone kolumny z tabel celem stworzenia stosownego raportu. Po zaprojektowaniu stworzono następujące raporty:

[illegible]

## Przychód w danym miesiącu w zależności od trasy

Odjazd	Przyjazd	1	10	11	12	2	3	4	5	6	7	8	9
Aleksandrow Kujawski	Aleksandrow Łódzki							103					
Aleksandrow Łódzki	Kalisz Pomorski								202		202	344	
Alwernia	Myslowice	210			298								
	Radzyn Chelminski		54					20					224
	Zakliczyn				18								
Andrychów	Jaraczewo					229	135	298		195			
	Oleszyce	35										163	
	Tluszc						380						422
Annapol	Izbica Kujawska					272	108						
	Jaraczewo				251								
	Pilawa			196									
Augustów	Bytom								142				
	Mrocza					78							
	Sulechów										107		
Babimost	Goleniów			117	321								
	Proszków												340

Imie	Nazwisko	Kursy Count
Ade	Niehues	2
Ahmar	Stowes	2
Aime	Brunck	4
Alexis	Acfalle	1
Alexandre	Resecker	2
Alf	Cackowski	3
Andoni	Sahakian	1
Andra	Ramjit	2
Antravious	Meris	4
Asael	Chrin	3
Baker	Ehrhorn	1
Benny	Dylewski	2
Bertin	Washuta	2
Brian	Ponko	1
Cecilio	Lige	1
Celedonio	Schloff	1
Celestine	Colosimo	4
Colter	Shiyu	1
Corgan	Toporowski	3
Davey	Salsberry	3

## Poziom wypełnienia pojazdów marki BMW w zależności od miesiący i ceny biletu

	1	10	11	12	2	3	4	5	6	7	8	9	
10		2,04			2,26	0,82	1,39	4,59	1,89	1,70	1,96		3,23
11		2,04	3,55	0,79	1,00		0,69	0,63	4,10	2,53	1,39	0,87	0,92
12		1,00	0,91	1,95	1,02	2,44	2,33	1,92		0,30	0,60	0,71	1,22
13			1,69	2,82	1,00	0,27	1,33		1,00	0,88	0,70	0,65	2,50
14		0,88	2,40	1,48	0,96		1,91	1,79	1,40	1,37	0,77		2,46

Tour Number	Ilosc Biletow Nor	Ilosc Biletow Ulg
10	4800	2400
12	4000	2000
3	4200	2200
5	4100	2300
7	4900	2800
9	4800	2600

	Aleksandrow Kujawski	Aleksandrow Łódzki	Alwernia	Andrychów
	Aleksandrow Łódzki	Kalisz Pomorski	Myslowice	Radzyn Chelmski
			Zakliczyn	Jaraczewo
				Oleszyce
				Tuszcza
2	24			19
3				6
4			29	2
5		17		
6				
7				
8			11	
9				19

[illegible]

## 7. Wnioski

Zadania wykonane w ramach projektu przebiegły pomyślnie. Jednym z podstawowych problemów do rozwiązania było stworzenie najlepszej struktury hurtowni danych tak, aby sprostała postawionym wymaganiom. Ma ona kluczowe znaczenie dla wydajności realizowanych zapytań oraz łatwości późniejszej pracy nad projektem. Stworzenie dobrej kostki analitycznej stanowi podstawę do generowania przejrzystych i użytecznych raportów. Stanowiła ona największe wyzwanie z faktu, iż proces debugowania potrafił przynosić liczne problemy związane na przykład z autoryzacją i rolami. Raporty dostarczają wielu informacji w zależności od ustawienia kolumn, dają też duże możliwości w sposobach wizualizacji danych, na przykład wykorzystując wykresy liniowe lub kołowe. Wykresy prezentują różnorodne możliwości w generowaniu raportów. W niektórych z nich konieczne było użycie składni języka MDX. Ograniczała się do dodania pól typu WHERE. W ostatnim zadaniu wygenerowano 5 raportów:

- Pierwszy określa liczbę kilometrów przejechanych przez kierowców w danej marce pojazdu. Pozwala zauważyć rodzaje aut wybierane najczęściej przez poszczególnych pracowników co może wpłynąć na dalsze zakupy pojazdów. Można też sprawdzić, ile kilometrów przejechał każdy kierowca, co pozwala oszacować jego wydajność.
- Drugi dotyczy przychodów na danej trasie w podziale na miesiące. Pozwala oszacować na jakiej trasie w zależności od sezonu generowany jest największy przychód oraz wpłynąć na zmiany dotyczące ilości kursów na przykład w sezonie zimowym i letnim. Powstał też w celu analizowania rentowności tras.
- Trzeci określa ilość kursów wykonanych przez kierowców, którzy podpisali umowę w roku 2000. Pozwala sprawdzić na przykład wydajność pracowników biorąc pod uwagę ich staż pracy w tej firmie. Raport może pomóc podjąć decyzję o redukcji personelu.
- Czwarty wyraża poziom wypełnienia pojazdów marki BMW w danych miesiącach biorąc pod uwagę cenę biletu. Raport ten przede wszystkim dostarcza informacji na temat tego jaką popularnością wśród pasażerów cieszy się dany model i jaki wpływ na wypełnienie ma cena biletu. Może się to przydać na przykład do zmiany ceny biletu, gdyby była za niska.

- Piąty pozwala oszacować ilość sprzedanych biletów w zależności od miesiąca. Podzielone są one na normalne i ulgowe. Pozwala zbadać na przykład to jaki wpływ ma okres wakacyjny na sprzedaż biletów, co za tym idzie zwiększyć lub zmniejszyć liczbę kursów.
- Szósty umożliwia przeanalizowanie sprzedaży biletów w zależności od trasy. Pozwala sprawdzić na jakich trasach jest jeździ najwięcej młodzieży na przykład celem wprowadzenia dodatkowych porannych/popołudniowych kursów.
- Siódmy raport pozwala sprawdzić wpływ mocy silników na czas przejazdu tras. W przypadku nikłej różnicy zakładając, że pojazdy z mocniejszym silnikiem są droższe można zredukować koszty na przyszłej flocie pojazdów.