INFORMATYKA – MÓJ SPOSÓB NA POZNANIE I OPISANIE ŚWIATA

PROGRAM NAUCZANIA INFORMATYKI Z ELEMENTAMI PRZEDMIOTÓW MATEMATYCZNO-PRZYRODNICZYCH

Informatyka – poziom podstawowy

Od chaosu do bazy danych

Andrzej Ptasznik

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

Tytuł: ***Od chaosu do bazy danych***

Autor: ***Andrzej Ptasznik***

Redaktor merytoryczny: ***prof. dr hab. Maciej M. Sysło***

Materiał dydaktyczny opracowany w ramach projektu edukacyjnego
Informatyka – mój sposób na poznanie i opisanie świata. Program nauczania informatyki z elementami przedmiotów matematyczno-przyrodniczych

www.info-plus.wwsi.edu.pl

infoplus@wwsi.edu.pl

Wydawca: Warszawska Wyższa Szkoła Informatyki
ul. Lewartowskiego 17, 00-169 Warszawa
www.wwsi.edu.pl
rektorat@wwsi.edu.pl

Projekt graficzny: *Marzena Kamasa*

Warszawa 2013

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska Wyższa Szkoła Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego

SCENARIUSZ TEMATYCZNY

OD CHAOSU DO BAZY DANYCH

INFORMATYKA – POZIOM PODSTAWOWY

OPRACOWANY W RAMACH PROJEKTU:

INFORMATYKA – MÓJ SPOSÓB NA POZNANIE I OPISANIE ŚWIATA.

PROGRAM NAUCZANIA INFORMATYKI

Z ELEMENTAMI PRZEDMIOTÓW MATEMATYCZNO-PRZYRODNICZYCH

Streszczenie

Podstawy relacyjnego modelu danych

Twórcą teorii relacyjnych baz danych jest Edgar Frank Codd. Podstawy relacyjnego modelu danych zostały opublikowane po raz pierwszy w 1970 roku w pracy *A Relational Model of Data for Large Shared Data Banks* (pol. Relacyjny model danych dla dużych współdzielonych banków danych). W pracy tej opisano podstawowe zależności, jakie mogą występować pomiędzy danymi oraz wprowadza główne założenia dotyczące modelu relacyjnego dla danych wraz z propozycją formalnych operatorów przeszukiwania danych. Burzliwy rozwój systemów opartych na relacyjnym modelu danych rozpoczął się wraz z wypuszczeniem na rynek przez firmę ORACLE w roku 1979 pierwszego komercyjnego relacyjnego systemu zarządzania bazą danych (Relational Database Management Systems, RDBMS). Sukces tego przedsięwzięcia zapoczątkował dominację baz danych stworzonych na modelu relacyjnym.

Relacyjne bazy danych zawdzięczają swój sukces bardzo prostemu sposobowi przechowywania danych – jest nim dwuwymiarowa tabela. Te zajęcia są w dużej części poświęcone przedstawieniu pojęcia tabeli relacyjnej.

Opis relacyjnego modelu danych można podzielić na trzy części:

- struktury danych – czyli sposób i zasady organizacji przechowywania danych oraz wytyczne, według których należy je projektować;
- języki manipulowania danymi – czyli jak zapisywać, modyfikować, usuwać oraz pobierać dane znajdujące się w bazie danych;
- integralność danych – czyli w jaki sposób zapewnić poprawność przechowywanych danych.

Wszystkie dane w bazie relacyjnej są przedstawione w postaci dwuwymiarowych tabel, zwanych relacjami. Projektowanie bazy danych sprowadza się do zdefiniowania grupy tabel opisujących dziedzinę, dla której chcemy utworzyć bazę danych. Intuicyjnie wiemy, że baza danych w banku będzie opisywana za pomocą innych tabel niż baza danych działająca w przychodni lekarskiej.

Tabela relacyjna

Przechowywanie danych w postaci dwuwymiarowych tabel wymaga ustalenia pewnych zasad, by dane tak składowane mogły być przydatne w bazach danych. Dla tabeli relacyjnej sformułowano pewne normy, które muszą być przestrzegane, żeby można ją było uznać za poprawną z punktu widzenia założeń modelu relacyjnego.

Spróbujmy przyjrzeć się niektórym z tych zasad.

Zasada 1

Każda tabela w bazie danych ma jednoznaczną nazwę.

Zasada ta nie powinna budzić wątpliwości, gdyż w przeciwnym przypadku nie można by było jednoznacznie identyfikować tabeli wśród wielu tabel o tej samej nazwie.

Zasada 2

Każda kolumna tabeli ma jednoznaczną nazwę w obrębie tej tabeli.

Powód i znaczenie tej zasady jest analogiczny jak zasady 1, ponieważ nazwy kolumn będą służyły do ich identyfikowania, a takie same nazwy uniemożliwiają jednoznaczną identyfikację kolumny.

Zasady 1 i 2 są oczywiste. Kolejna reguła głębiej wnika w istotę relacyjnego modelu danych.

Zasada 3

Wszystkie wartości w kolumnie są tego samego typu.

Jak już wcześniej powiedzieliśmy, kolumna w tabeli służy do przechowywania danych o jednej cesze opisywanych w tabeli obiektów. Zatem dane w kolumnie powinny być tego samego typu.

Spróbujmy wyjaśnić to na bazie przykładu. Jeżeli w pewnej tabeli istnieje kolumna o nazwie DataUrodzenia, to znaczy, że chcemy tam przechowywać dane określające datę i wszystkie wartości zapisane w takiej kolumnie muszą mieć postać poprawnie zapisanej daty. W innym przypadku będziemy mieli poważne problemy z wykorzystaniem i interpretacją zapisanych danych.

W zależności od rodzaju danych, rozróżniamy różne ich typy:

- Dane znakowe – przykładowo:
  - Nazwisko,

- Imię,
- Opisy.

➠ Liczby całkowite – przykładowo:

- Liczba dzieci,
- Wzrost w cm,
- Stan towaru w magazynie.

➠ Liczby ułamkowe – przykładowo:

- Kwoty wpłaty,
- Wartość faktury,
- Wartość rabatu.

➠ Daty – przykładowo:

- Data urodzenia,
- Data zatrudnienia,
- Termin przydatności do spożycia.

Nieprzestrzeganie tej zasady mogłoby doprowadzić do sytuacji pokazanej na rysunku 1. Przykładowa tabela spełniająca ten postulat modelu relacyjnego zaprezentowana jest na rysunku 2.

Osoby					
Nazwisko	**Imię**	**DataUrodzenia**	**Pesel**	**Waga**	**Wzrost**
Zagłoba	Onufry	12.08.1977	76091234532	92,5	176
Podbipięta	Longinus	22 styczeń 1954	Pesel:54012276562	56 i pół	1,95
Dulska	Aniela	Dziewiąty maj 65 r.	65050934aaaa	68 i 600 g.	182 i 50/100
Kmicic	Andrzej	Kwiecień 1986 rok	Nie znany	Około 76	Prawie 190

Rysunek 1. Tabela z danymi różnych typów zapisanych w jednej kolumnie.

Osoby					
Nazwisko	**Imię**	**DataUrodzenia**	**Pesel**	**Waga**	**Wzrost**
Zagłoba	Onufry	12.08.1977	77081276534	92,5	176
Podbipięta	Longinus	22.01.1922	22012276575	74,3	205
Dulska	Aniela	09.04.1943	43040945645	68,6	159
Kmicic	Andrzej	21.11.1992	92112176478	77,0	187

Rysunek 2. Przykładowa tabela z danymi poprawnych typów danych.

Zasada 4

W tabeli nie mogą istnieć dwa identyczne wiersze, wiersze są różne, tabela może nie zawierać wierszy.

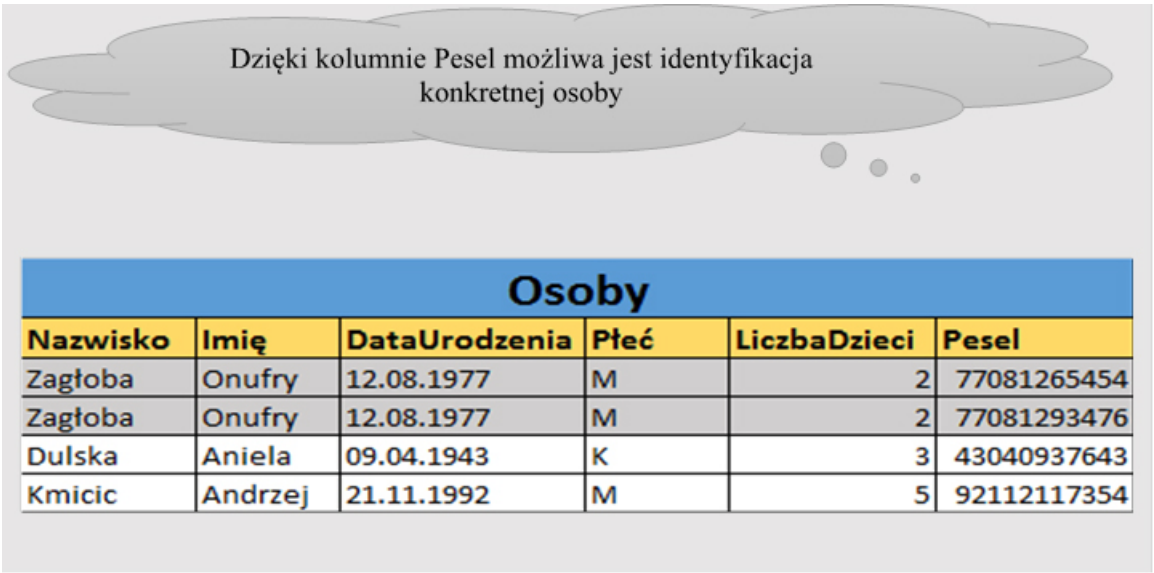
Zasada 4 prezentuje dwie tezy – jedną, która mówi, że w tabeli relacyjnej nie mogą znajdować się dwa identyczne wiersze, i drugą, która informuje, że może istnieć tabela relacyjna, która nie zawiera, w danym momencie, żadnych wierszy. Zasada ta jest bardzo istotna z punktu widzenia modelu relacyjnego, gdyż bezpośrednio z niej wynika konieczność znajdowania się w tabeli relacyjnej klucza podstawowego.

Omówimy istotę tej zasady na przykładzie pokazanym na rysunku 3. Dwa pierwsze wiersze w tabeli są identyczne (teoretycznie mogą istnieć dwie osoby opisywane takimi samymi danymi), a tym samym przeczą jednej z podstawowych zasad tabeli relacyjnej. Z omawianej zasady jednoznacznie wynika, że w tabeli relacyjnej musi znajdować się kolumna (lub zbiór kolumn), która dla każdego wiersza przyjmuje inną wartość. Taką kolumnę nazywamy **kluczem podstawowym** tabeli (*primary key*).

Osoby				
Nazwisko	**Imię**	**DataUrodzenia**	**Płeć**	**LiczbaDzieci**
Zagłoba	Onufry	12.08.1977	M	2
Zagłoba	Onufry	12.08.1977	M	2
Dulska	Aniela	09.04.1943	K	3
Kmicic	Andrzej	21.11.1992	M	5

Rysunek 3. Tabela zawierająca dwa identyczne wiersze.

Dodając w tabeli na rysunku 3 dodatkową kolumnę Pesel (każda osoba ma inny numer Pesel) uzyskujemy klucz podstawowy, a tym samym tabela spełnia zasadę 4. Wynik tej operacji jest pokazany na rysunku 4.



Osoby					
Nazwisko	**Imię**	**DataUrodzenia**	**Płeć**	**LiczbaDzieci**	**Pesel**
Zagłoba	Onufry	12.08.1977	M	2	77081265454
Zagłoba	Onufry	12.08.1977	M	2	77081293476
Dulska	Aniela	09.04.1943	K	3	43040937643
Kmicic	Andrzej	21.11.1992	M	5	92112117354

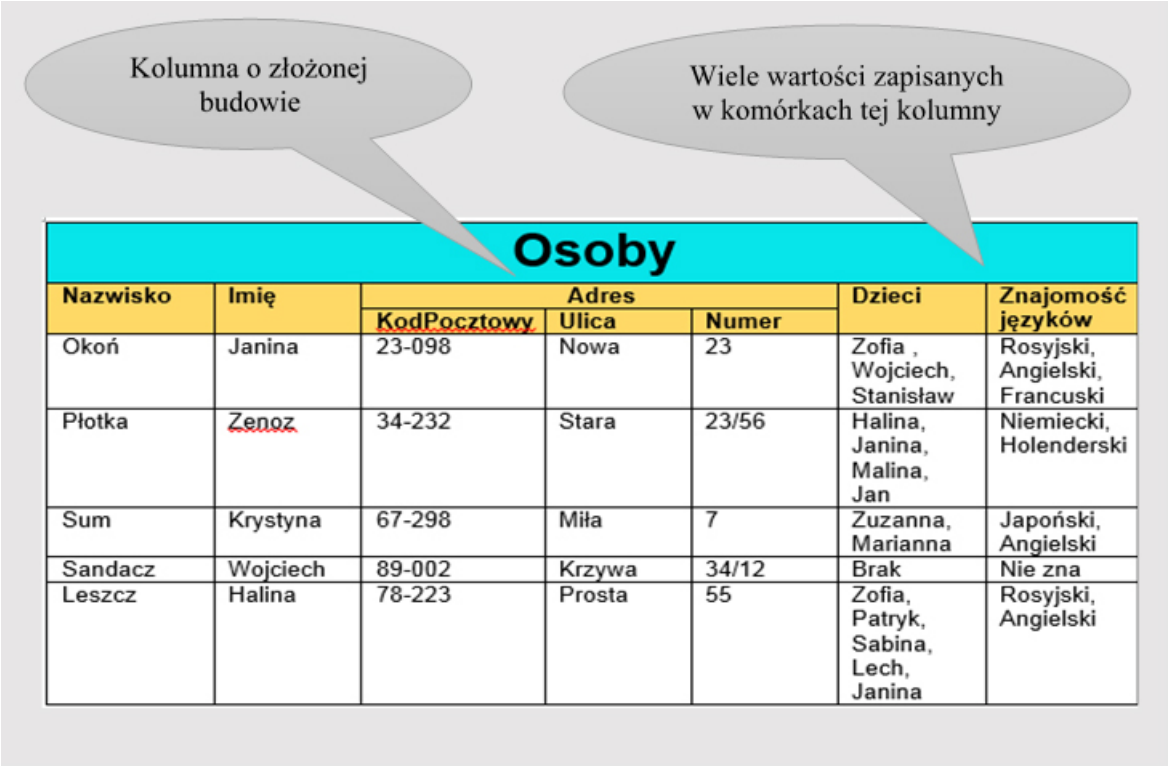
Rysunek 4. Poprawna tabela relacyjna.

Dodanie klucza podstawowego spowodowało, że pozornie dwa identyczne wiersze jednak się różnią i są opisami dwóch różnych osób. Z omawianego przykładu wynika fakt, że rozróżnianie wierszy w tabeli relacyjnej musi opierać się na różnicy danych. Klucz podstawowy jest wykorzystywany do jednoznacznego wskazywania wiersza, ponieważ tylko jeden wiersz może mieć określoną wartość klucza. Istnienie kolumny klucza podstawowego jest jednym z filarów relacyjnego modelu danych. W tym miejscu nie będziemy prowadzić dyskusji, czy numer Pesel jest dobrym „kandydatem" na klucz podstawowy, ale w wielu sytuacjach mogłyby zaistnieć problemy z tak wybraną kolumną kluczową choćby w przypadku, kiedy w naszej tabeli trzeba by zapisać dane obcokrajowca.

Zasada 5

W tabeli relacyjnej przechowywane są dane oparte na prostych typach danych (dane elementarne).

Sytuacją niepożądaną z punktu widzenia tej zasady tabeli relacyjnej jest zapisywanie wielu danych w jednej komórce tabeli. Tabela przedstawiona na rysunku 5 nie spełnia tak sformułowanej zasady, ponieważ w niektórych kolumnach przechowuje po kilka wartości oraz dodatkowo niektóre kolumny mają złożoną strukturę (kolumna Adres).



Osoby

Nazwisko	Imię	Adres			Dzieci	Znajomość języków
		KodPocztowy	Ulica	Numer		
Okoń	Janina	23-098	Nowa	23	Zofia , Wojciech, Stanisław	Rosyjski, Angielski, Francuski
Płotka	Zenoz	34-232	Stara	23/56	Halina, Janina, Malina, Jan	Niemiecki, Holenderski
Sum	Krystyna	67-298	Miła	7	Zuzanna, Marianna	Japoński, Angielski
Sandacz	Wojciech	89-002	Krzywa	34/12	Brak	Nie zna
Leszcz	Halina	78-223	Prosta	55	Zofia, Patryk, Sabina, Lech, Janina	Rosyjski, Angielski

Rysunek 5. Tabela o nieprawidłowej strukturze (nie spełnia zasady 5).

Zapisanie w jednej kolumnie danego wiersza wielu wartości mogłoby sprawiać duże problemy w trakcie wykorzystywania tak zapisanych danych. W dalszej części, przy omawianiu przykładowego projektu bazy danych zostanie pokazany sposób rozwiązania tego problemu. Kolumny o nazwie Dzieci i ZnajomośćJęzyków przechowują zbiory wartości, czyli nie są to typy proste.

Zasada 6

Kolejność wierszy i kolejność kolumn w tabeli relacyjnej nie mają znaczenia.

Zasada 6 wynika z faktu, że model relacyjny opiera się na pojęciu zbioru, a w zbiorach nie jest istotna kolejność elementów. Na rysunku 6 pokazano trzy przykładowe tabele, które pozornie różnią się od siebie (inna kolejność wierszy i inna kolejność kolumn), ale z punktu widzenia modelu relacyjnego jest to dokładnie ten sam zestaw danych.

Osoby				
Nazwisko	**Pesel**	**DataUrodzenia**	**Imię**	**Wzrost**
Podbipięta	22012276575	22.01.1922	Longinus	205
Kmicic	92112176478	21.11.1992	Andrzej	187
Zagłoba	77081276534	12.08.1977	Onufry	176
Dulska	43040945645	09.04.1943	Aniela	159

Osoby				
Nazwisko	**Pesel**	**DataUrodzenia**	**Imię**	**Wzrost**
Dulska	43040945645	09.04.1943	Aniela	159
Kmicic	92112176478	21.11.1992	Andrzej	187
Zagłoba	77081276534	12.08.1977	Onufry	176
Podbipięta	22012276575	22.01.1922	Longinus	205

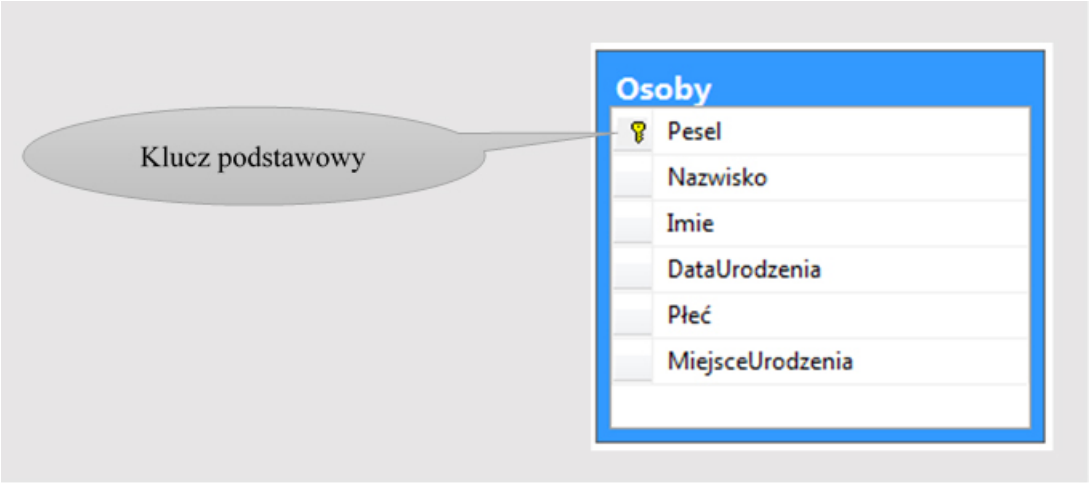
Osoby				
Nazwisko	**Imię**	**DataUrodzenia**	**Pesel**	**Wzrost**
Zagłoba	Onufry	12.08.1977	77081276534	176
Podbipięta	Longinus	22.01.1922	22012276575	205
Dulska	Aniela	09.04.1943	43040945645	159
Kmicic	Andrzej	21.11.1992	92112176478	187

Rysunek 6. Przykładowe tabele z identycznym zestawem danych.

Omówione zasady określają podstawowe własności tabeli, zgodnie z zasadami modelu relacyjnego. W dalszej części omawiamy podstawy projektowania baz danych. Opisując projekt bazy danych będziemy używali innej notacji struktury tabeli. Na rysunku 7 pokazano klasyczny układ tabeli (służący prezentowaniu zawartości tabeli – wiersze danych), a na rysunku 8 postać notacji, którą będziemy od tej pory używać do prezentowania projektu bazy danych.

Osoby					
Pesel	Nazwisko	Imie	DataUrodzenia	Płeć	MiejsceUrodzenia

Rysunek 7. Struktura przykładowej tabeli – postać klasyczna.



Rysunek 8. Struktura przykładowej tabeli – struktura projektu bazy danych.

Podsumowując, zasady które muszę spełniać tabele relacyjne są dość oczywiste. Jest to jeden z powodów sukcesu relacyjnego modelu danych. W dalszej części zajęć poznamy podstawy projektowania relacyjnych baz danych.

Czas realizacji

3 x 45 minut

Tematy lekcji

1 Porządkowanie chaosu
2 Tworzenie bazy danych i definiowanie tabel
3 Definiowanie ograniczeń

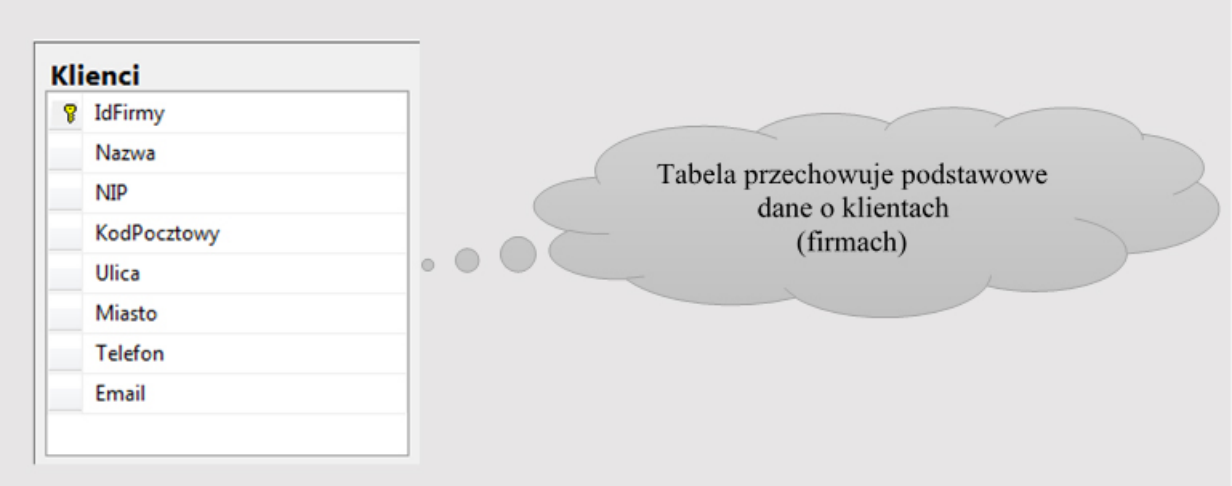
LEKCJA NR 1

TEMAT: Porządkowanie chaosu

Wprowadzenie do lekcji

Analiza wybranego projektu bazy danych

Wyobraźmy sobie sytuację, że w pewnej bazie danych istnieje tabela o nazwie `Klienci`, której strukturę pokazano na rysunku 9.



Rysunek 9. Struktura tabeli o nazwie **`Klienci`**.

Na pierwszy rzut oka tabela wygląda prawidłowo.

Przykładowa zawartość takiej tabeli mogłaby wyglądać tak, jak pokazano na rysunku 10.

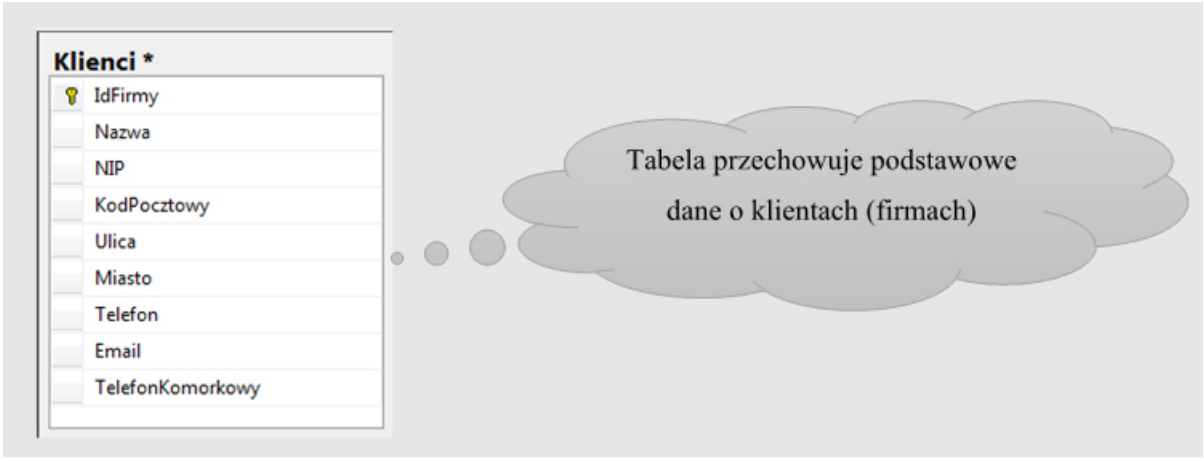
Klienci

IdKlienta	Nazwa	NIP	KodPocztowy	Ulica	Miasto	Telefon	Email
1	Alfa sp. z o.o.	8349221365	91-832	Nowa 5	Opole	505 787 456	biuro@alfa.pl
2	Beta S.A.	7169123549	33-126	Stara 7	Gliwice	711 767 175	NULL
3	OmegaSoft sp. z o.o.	7183485195	44-612	Długa 123	Leszno	NULL	NULL

Rysunek 10. Przykładowa zawartość tabeli o nazwie **`Klienci`**.

W dalszym ciągu nie widać problemów – zapisy w tabeli wyglądają zupełnie sensownie. Uwagę zwraca jedynie brak pewnych danych, co jest oznaczane wartością `NULL` – nie jest to błąd, taka wartość jest dopuszczalna w bazach danych, chociaż nie zawsze jest pożądana.

Może pojawić się problem, gdy użytkownicy takiej bazy danych będą natrafiać na sytuacje, trudne do przewidzenia na etapie projektowania. Chcieliby, na przykład, przechowywać numer telefonu komórkowego. Można to rozwiązać bardzo prosto dodając do tabeli nową kolumnę przeznaczoną do przechowywania numerów telefonów komórkowych. Zmodyfikowana tabela może mieć strukturę pokazaną na rysunku 10.



Rysunek 11. Struktura tabeli o nazwie **`Klienci`** po modyfikacji.

Struktura tej tabeli wskazuje, że problem został rozwiązany, ponieważ w zmodyfikowanej strukturze tabeli jest miejsce na zapisanie danych o numerze telefonu komórkowego. Przykładową zawartość tabeli o nazwie `Klienci` ukazuje rysunek 12.

Klienci								
IdKlienta	Nazwa	NIP	KodPocztowy	Ulica	Miasto	Telefon	Email	TelefonKomorkowy
1	Alfa sp. z o.o.	8349221365	91-832	Nowa 5	Opole	505 787 456	biuro@alfa.pl	NULL
2	Beta S.A.	7169123549	33-126	Stara 7	Gliwice	711 767 175	NULL	NULL
3	OmegaSoft sp. z o.o.	7183485195	44-612	Długa 123	Leszno	42 88 34654	prezes@omega.	501 187 046

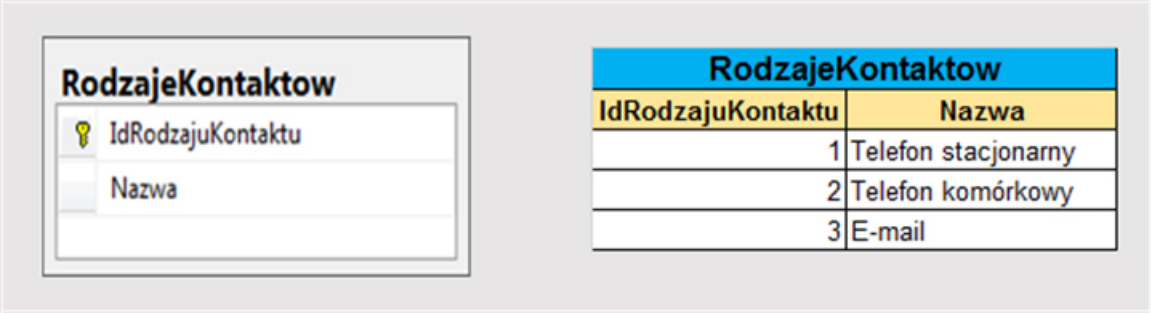
Rysunek 12. Przykładowa zawartość tabeli o nazwie **`Klienci`** po modyfikacji.

Problem został tylko pozornie rozwiązany, gdyż dodanie nowej kolumny do tabeli w już istniejącej i działającej bazie danych nie jest sprawą tak prostą, jak zaprezentowano. Pozostaje jeszcze pytanie, czy ta modyfikacja rozwiązuje problem na dłużej, czy mamy pewność, że w trakcie dalszej eksploatacji tej bazy danych nie będzie trzeba dodać kolejnych kolumn, na przykład w celu zapisania większej liczby numerów telefonu albo adresu strony www, numeru faksu, numeru GG itd.

Widać wyraźnie, że zaproponowane przez nas rozwiązanie nie jest trwałe i elastyczne. W trakcie eksploatacji bazy danych może się bowiem okazać, że nasze tabele nie są w stanie zapisać danych, które są potrzebne użytkownikom.

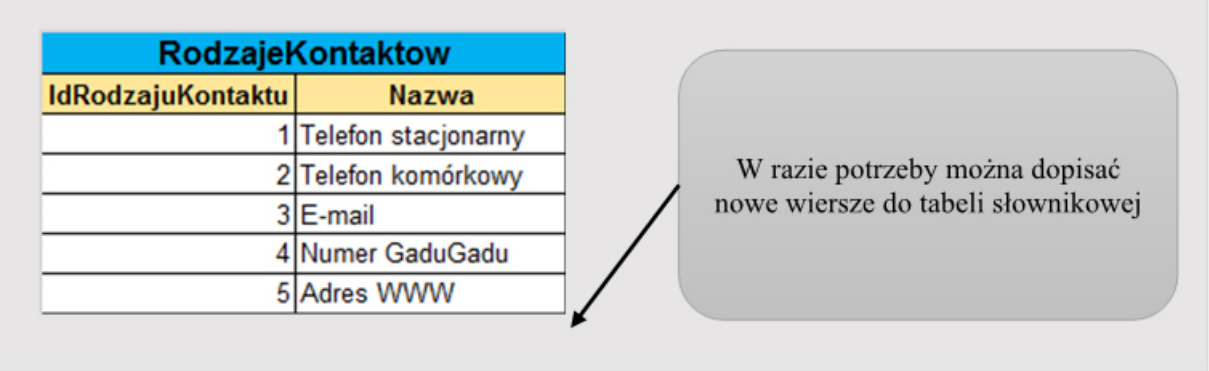
Chwila zastanowienia pozwala rozwiązać ten problem raz a dobrze. Widać, że trudnością staje się ustalenie, jakie dane mogą być nam potrzebne. Proponujemy utworzenie tabeli, która będzie takie dane przechowywała.

Na rysunku 13 pokazano strukturę i przykładową zawartość takiej tabeli.



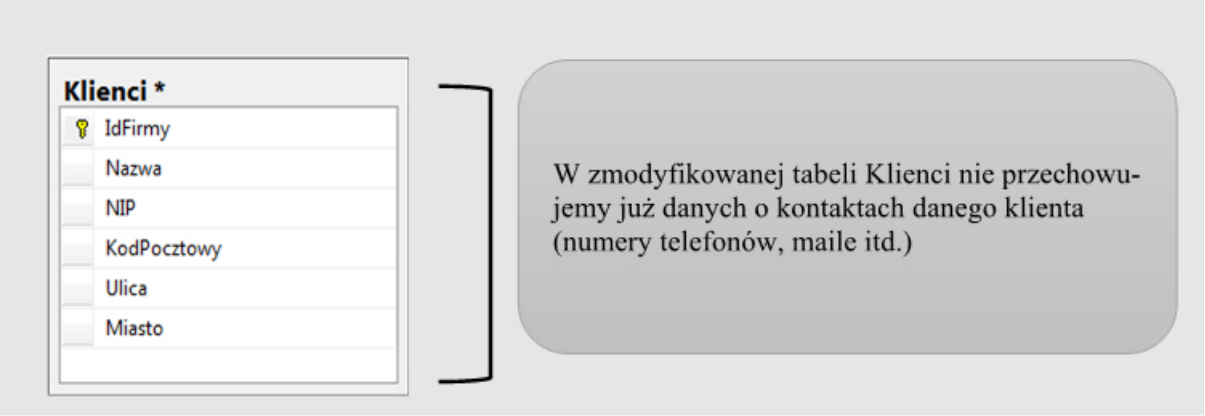
Rysunek 13. Struktura i przykładowa zawartość tabeli słownikowej o nazwie **`RodzajeKontaktow`**.

W przypadku konieczności przechowywania w bazie danych informacji o numerach Gadu Gadu i adresach stron WWW – to wystarczy dopisać kolejne wiersze do tabeli i otrzymamy tabelę z odpowiadającą nam zawartością. Tym razem nie wymaga to dodania nowych kolumn do istniejącej tabeli, co jest czynnością trudną i złożoną w działającej bazie danych, a jedynie dopisania nowego wiersza lub wierszy do istniejącej tabeli, a to jest czynnością całkowicie naturalną. Na rysunku 14 pokazana jest tabela o nazwie `RodzajeKontaktow` z uzupełnionymi wpisami.



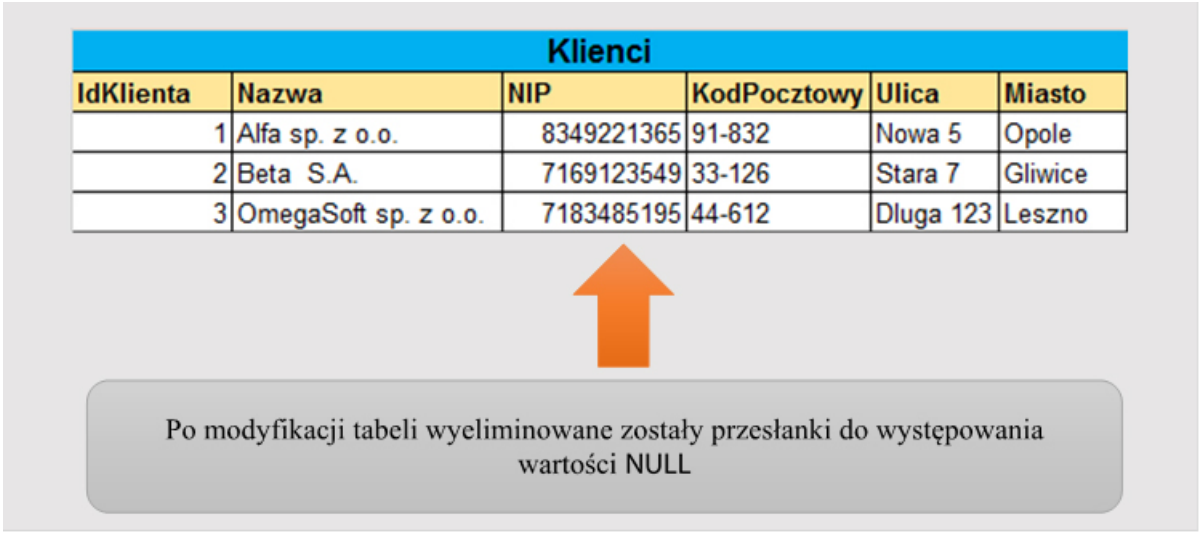
Rysunek 14. Uzupełniona zawartość tabeli o nazwie **`RodzajeKontaktow`**.

Wykonaliśmy pierwszy krok, ale do rozwiązania problemu droga jeszcze daleka. Kolejny etap to modyfikowanie tabeli `Klienci`, przez usunięcie z niej wszystkich kolumn przechowujących dane o kontaktach. Strukturę zmodyfikowanej tabeli o nazwie `Klienci` prezentuje rysunek 15.



Rysunek 15. Zmodyfikowana struktura tabeli o nazwie **`Klienci`**.

Tabela z rysunku 15 została trochę odchudzona i tym samym uproszczona, ale w dalszym ciągu nie mamy miejsca na zapisywanie danych o kontaktach konkretnych klientów. Przykładową zawartość tej tabeli pokazano na rysunku 16.

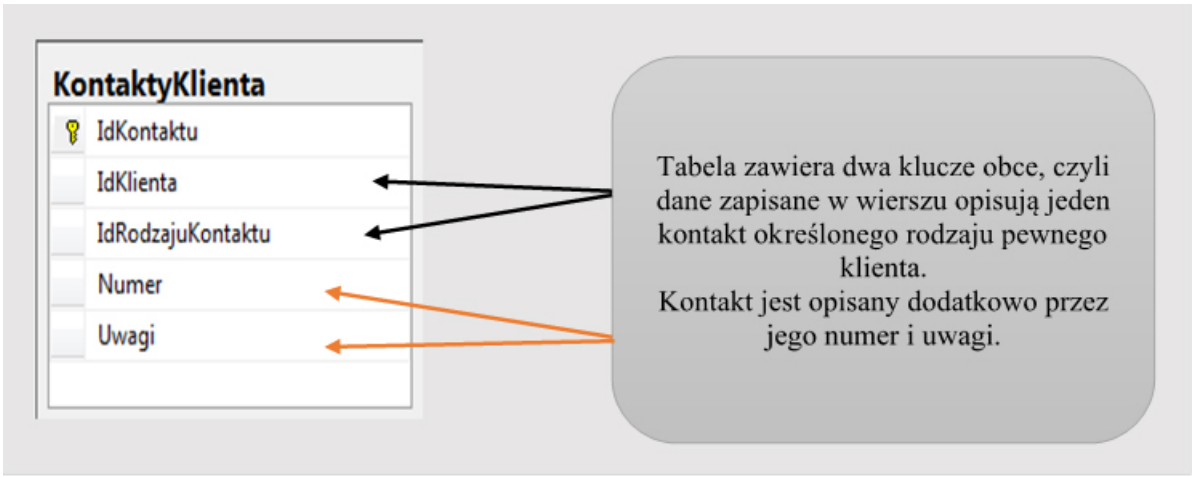


Rysunek 16. Zmodyfikowana struktura tabeli o nazwie **`Klienci`**.

Ostatnią tabelą, którą musimy utworzyć, jest tabela przeznaczona do przechowywania danych o kontaktach. Dane zapisane w takiej tabeli powinny zawierać uzupełnienie poniższego zdania:

„Z pewnym klientem (idklienta) istnieje pewien rodzaj kontaktu (idRodzajuKontaktu), z którym związany jest numer (numer – rozumiany, jako numer telefonu, GG, adres e-mail itp.) oraz pewne dodatkowe uwagi (uwagi)".

Struktura tabeli odpowiadającej takim wymaganiom została pokazana na rysunku 17. Tabelę tego typu nazywamy tabelą powiązań (asocjacyjną).



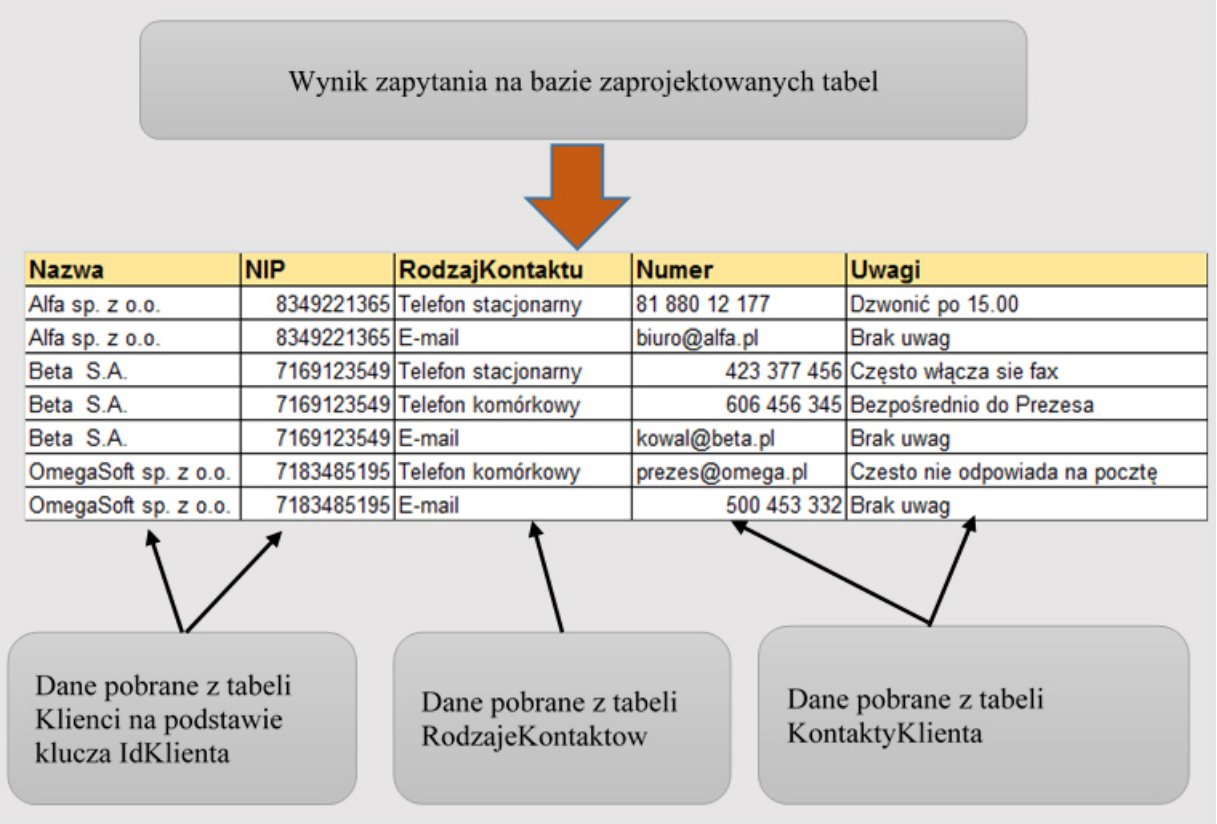
Rysunek 17. Struktura tabeli o nazwie **`KontaktyKlienta`**.

Przykładową zawartość takiej tabeli uwidacznia kolejny rysunek. Na pierwszy rzut oka dane zapisane w tej tabeli wydają się mało czytelne, ale pamiętajmy o tym, że dzięki kluczom obcym (idklienta, idRodzajuKontaktu) mamy dostęp do dodatkowych danych zapisanych w powiązanych tabelach. Przykładową zawartość tabeli o nazwie `KontaktyKlienta` pokazano na rysunku 18.

KontaktyKlienta				
IdKontaktu	IdKlienta	IdRodzajuKontaktu	Numer	Uwagi
1	1	1	81 880 12 177	Dzwonić po 15.00
2	1	3	biuro@alfa.pl	Brak uwag
4	2	1	423 377 456	Często włącza sie fax
5	2	2	606 456 345	Bezpośrednio do Prezesa
6	2	3	kowal@beta.pl	Brak uwag
7	3	2	prezes@omega.pl	Czesto nie odpowiada na pocztę
9	3	3	500 453 332	Brak uwag

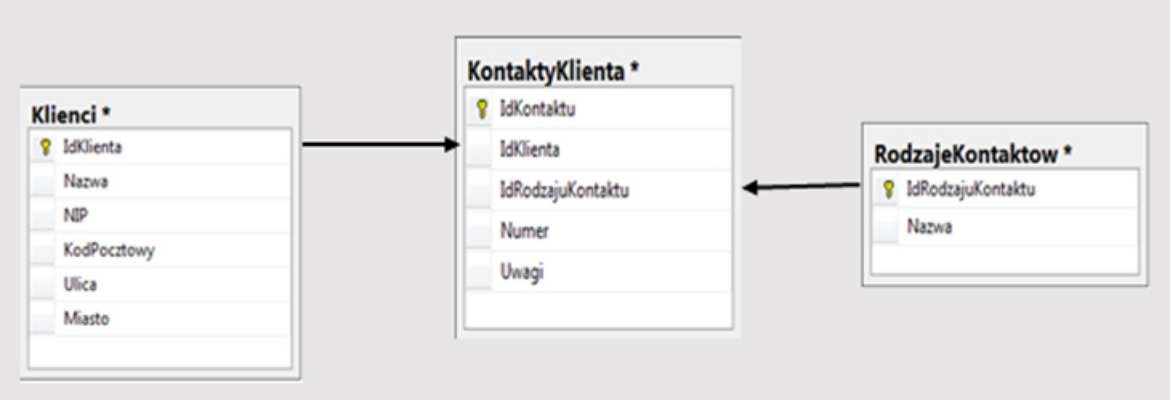
Rysunek 18. Przykładowa zawartość tabeli o nazwie **`KontaktyKlienta`**.

A jak wyglądałyby dane z tej tabeli po zinterpretowaniu ich z wykorzystaniem kluczy obcych? Zadanie takie w bazach danych jest realizowane poprzez odpowiednie zapytania. Chociaż nie wiemy jeszcze, w jaki sposób je realizować, to na rysunku 19 pokazano wynik zapytania przeprowadzony na pokazanych wcześniej przykładowych danych.



Rysunek 19. Wynik zapytania dla przykładowych danych.

Teraz zaprezentowane dane są czytelniejsze i w ten sposób przekształcono je w pewną informację. Poniżej widzimy fragment projektu, w którym zamiast jednej tabeli `Klienci` są trzy tabele (rysunek 20).



Rysunek 20. Schemat bazy danych.

Pojawia się pytanie, czy „gra warta była świeczki", ale odpowiedź na takie pytanie wcale nie jest jednoznaczna. Z jednej strony poprawiliśmy projekt, który teraz jest bardziej elastyczny, a z drugiej – stał się bardziej złożony i jego obsługa, przy korzystaniu z bazy danych, też będzie bardziej skomplikowana. Tego typu pytania są codziennością w trakcie projektowania i eksploatowania baz danych. Zwykle, gdy poprawiamy jeden aspekt rozwiązania, to na ogół kosztem innego, zatem podstawowym wyzwaniem etapu projektowania jest wybór odpowiedniego kompromisu.

Podstawa programowa

Etap edukacyjny: IV, przedmiot: informatyka (poziom podstawowy)

Cele kształcenia – wymagania ogólne

I.Wyszukiwanie, gromadzenie i przetwarzanie informacji z różnych źródeł; opracowywanie za pomocą komputera: rysunków, tekstów, danych liczbowych, motywów, animacji, prezentacji multimedialnych.

Treści nauczania – wymagania szczegółowe

4. Opracowywanie informacji za pomocą komputera, w tym: rysunków, tekstów, danych liczbowych, animacji, prezentacji multimedialnych i filmów. Uczeń:
   6) tworzy bazę danych, posługuje się formularzami, porządkuje dane, wyszukuje informacje, stosując filtrowanie;
   7) wykonuje podstawowe operacje modyfikowania i wyszukiwania informacji na relacyjnej bazie danych.

Cel

Podstawowym celem lekcji jest wyjaśnienie istoty relacyjnego modelu danych oraz zapoznanie uczniów z elementami środowiska MS SQL Server 2012.

Słowa kluczowe

relacyjny model danych, tabela relacyjna, klucz podstawowy, klucz obcy

Co przygotować

- Zainstalować MS SQL Server 2012 Express Edition – opis procesu instalacji opisany jest w pliku dodatkowym o nazwie „Instalacja pakietu MS SQL Server 2012 Express Edition With Advanced Services". Edycja Express systemu SQL Server 2012 jest ogólnie dostępną darmową wersją oprogramowania do wszystkich zastosowań (także komercyjnych). Korzystanie z tego oprogramowania pozwala zapoznać uczniów z profesjonalnym Systemem Zarządzania Bazami Danych i ułatwia nauczanie języka SQL. Do realizacji scenariusza można także wykorzystać program Access. Podstawowym problemem związanym z wykorzystaniem tego programu jest jego dostępność w najnowszej wersji.

  Wprowadzenie do cyklu lekcji scenariusza omówić z wykorzystaniem prezentacji, zawartej w plikach dodatkowych, o nazwie „Proces normalizacji danych".

Przebieg zajęć

Wprowadzenie (30 minut)

Nauczyciel wykorzystuje prezentację „Proces normalizacji danych". Szczególną uwagę należy zwrócić na fakt, że w modelu relacyjnym jedynym elementem opisu dziedziny problemu jest dwuwymiarowa tabela, natomiast na etapie projektowania bazy danych (normalizacja) staramy się wyeliminować redundancję, a powiązanie danych zapisanych w różnych tabelach zapewniamy dzięki kluczom obcym.

Dyskusja podsumowująca (10 minut)

W ramach dyskusji należy zaproponować uczniom przygotowanie w ramach zadania dodatkowego projektu bazy danych dla wybranej dziedziny problemu (np. biblioteka, wystawianie ocen itp.). Prace nad projektem bazy danych należy prowadzić w grupach ok. 5-osobowych.

Sprawdzenie wiedzy

Podstawowe sprawdzenie wiedzy można zrealizować na podstawie testu.

Ocenianie

Oceny na podstawie testu oraz pracy grupowej przy realizacji projektu i jej wyników.

Dostępne pliki

1. Opis procesu instalacji MS SQL Server 2012 – Instalacja pakietu MS SQL Server 2012 Express Edition With Advanced Services.docx – materiały pomocnicze 1
2. Uwagi do realizacji scenariusza z wykorzystaniem programu Access.docx (podane na zakończenie scenariusza oraz w materiałach pomocniczych 2)
3. Prezentacja
4. Filmy instruktażowe:

   Film 1: Komentarz do procesu instalacji

   Film 2: Podsumowanie procesu instalacji
5. Test 1

LEKCJA NR 2

TEMAT: Tworzenie bazy danych i definiowanie tabel

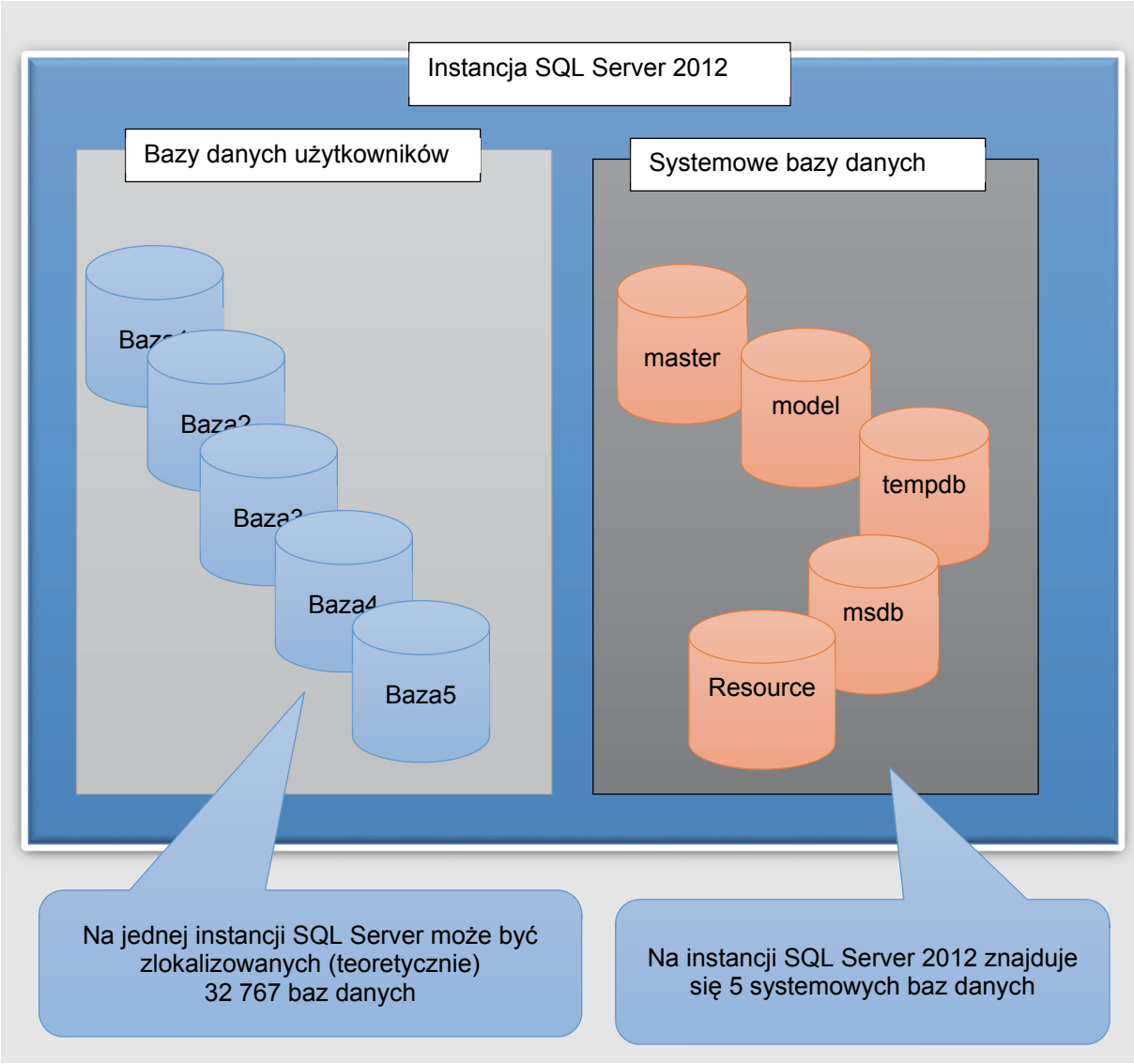
Streszczenie

Tworzenie własnej bazy danych

Zainstalowana i skonfigurowana instancja serwera SQL Server 2012 jest gotowa do tworzenia baz danych. SQL Server 2012 jest Systemem Zarządzania Relacyjnymi Bazami Danych, z pomocą którego można tworzyć i eksploatować gigantyczne zasoby danych. Przytoczymy kilka parametrów (ograniczeń), aby uzmysłowić sobie potencjał serwera SQL Server 2012:

- Na jednej instancji serwera SQL Server 2012 może być zlokalizowanych 32 767 baz danych,
- Jedna baza danych może mieć wielkość 524 272 terabajtów.

Na rysunku 21 pokazano podstawową architekturę instancji SQL Server 2012.



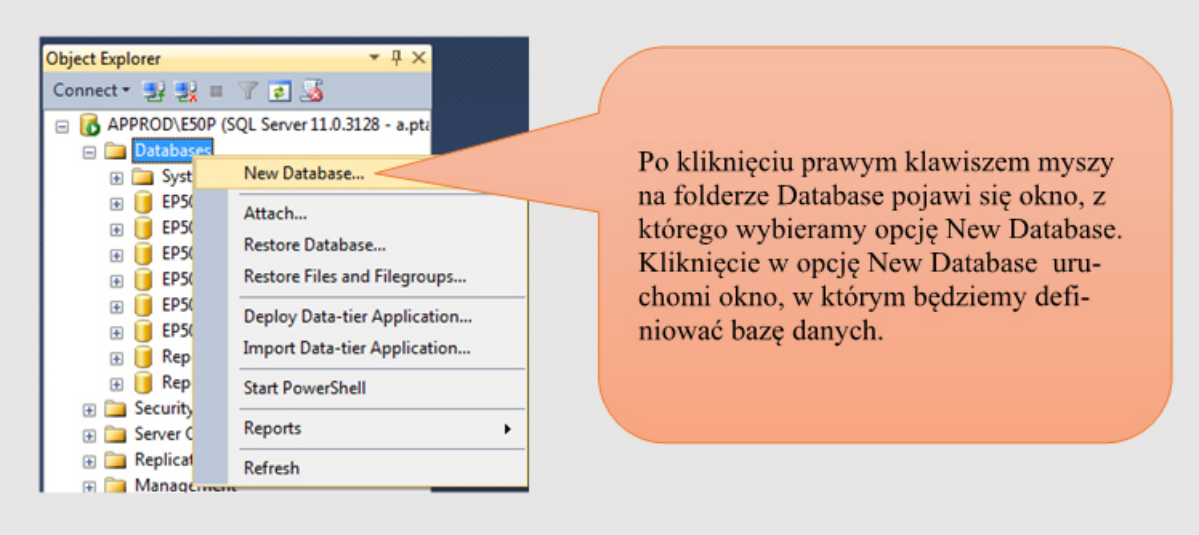
Rysunek 21. Architektura instancji SQL Server 2012.

Trudno sobie wyobrazić tak duże ilości danych, żeby wyczerpać cytowane wyżej ograniczenia, ale dwadzieścia kilka lat temu trudno było sobie wyobrazić bazę danych o wielkości 1 gigabajta.

Na instancji serwera znajdują się także systemowe bazy danych, czyli takie bazy, które SQL Server 2012 wykorzystuje do obsługi swojego działania.

Przystąpimy teraz do krótkiego omówienia procesu tworzenia własnej bazy danych. Przed tworzeniem bazy danych powinniśmy zastanowić się nad kilkoma elementami, od których może zależeć ustawienie parametrów konfiguracyjnych. Proces kreowania bazy danych może być bardzo prosty wtedy, gdy decydujemy się utworzyć bazę danych skonfigurowaną domyślnie. W takiej sytuacji jedynym zadaniem jest ustalenie jej nazwy (każda baza danych na instancji serwera musi mieć unikatową nazwę).

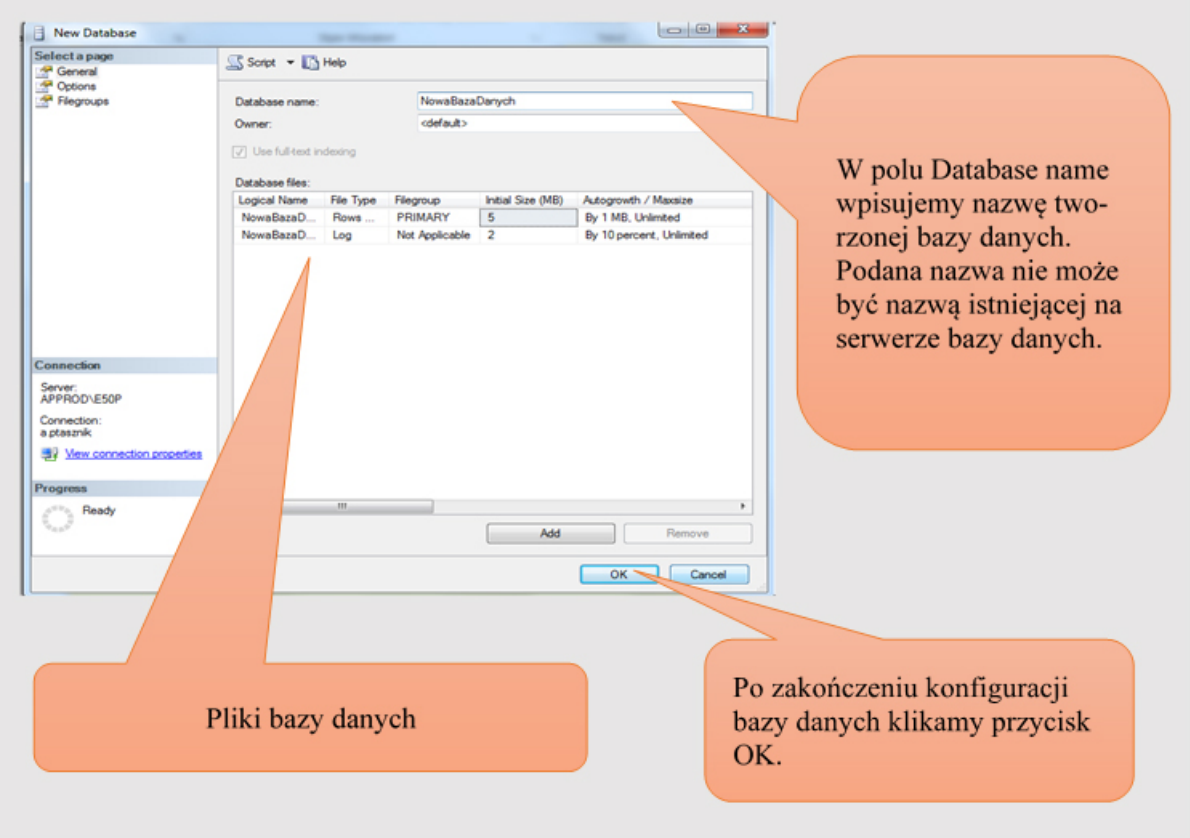
Na rysunku 22 pokazano wybór opcji inicjującej proces tworzenia nowej bazy danych w środowisku SQL Server Management Studio.



Rysunek 22. Wybór opcji New Database.

W tym miejscy warto przez chwilę zastanowić się, czym będzie nasza baza po jej utworzeniu. W nowo utworzonej bazie danych nie będzie tabel, które wcześniej zaprojektowaliśmy, gdyż dopiero wtedy gdy baza danych istnieje można w niej tworzyć tabele i inne obiekty bazy danych.

Język **SQL** (ang. *Structured Query Language*) został stworzony do pracy z relacyjną bazą danych. Jest to język nieproceduralny, należący do grupy języków deklaratywnych. Składnia języka SQL opisuje, co ma być zrobione, a nie jak należy to wykonać. Problem „jak wykonać" przeniesiony został na poziom systemu zarządzania relacyjną bazą danych. Pierwowzór języka SQL – SEQUEL (Structured English Query Language) – został zaprojektowany przez IBM w 1974 roku. Pierwsza zaś wersja SQL została komercyjnie zastosowana przez firmę Oracle Corporation w roku 1979. Do dnia dzisiejszego język SQL jest ciągle rozwijany i znajduje zastosowanie w większości systemów opartych na relacyjnym modelu danych. W założeniach twórców miał to być język uniwersalny, możliwie prosty i maksymalnie zbliżony do języka naturalnego. Czy to założenie zostało zrealizowane trudno jednoznacznie ocenić, ponieważ jest to język łatwy i prosty... ale nie zawsze. Na rysunku 23 pokazano postać okna, w którym możemy zdefiniować i skonfigurować naszą bazę danych.



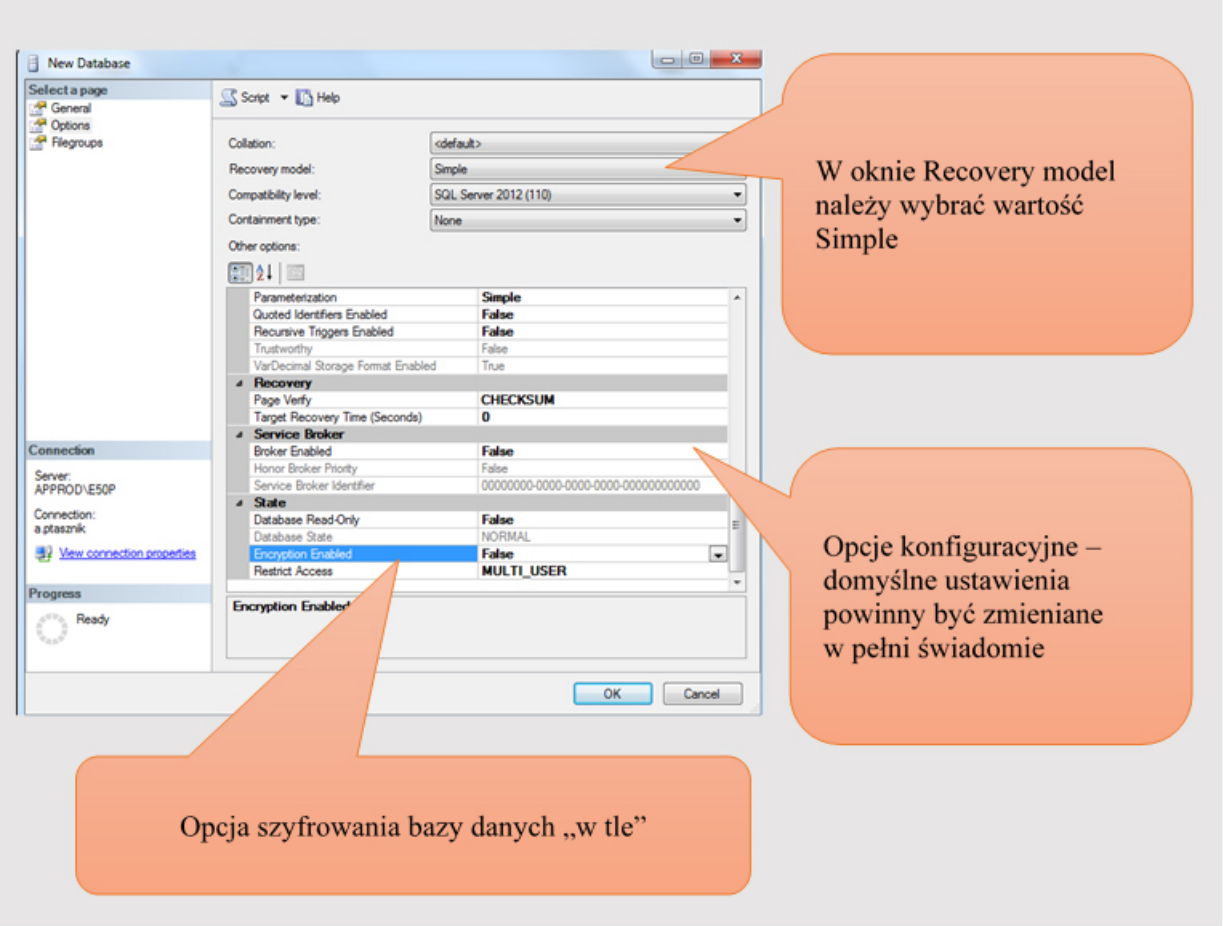
Rysunek 23. Postać okna New Database.

Kiedy nie stawiamy bazie danych szczególnych wymagań, po wpisaniu w oknie Database Name ustalonej nazwy dla nowo tworzonej bazy danych i kliknięciu przycisku OK, na serwerze zostanie utworzona baza danych według domyślnych ustawień parametrów. Domyślna parametryzacja bazy danych nie jest błędem, tym bardziej że producent zapewnia ustawienie tych parametrów według strategii najlepszych praktyk. W procesie definiowania nowej bazy danych możemy zmienić ustawienia domyślne w zakresie:

- rozmiaru wyjściowego bazy danych,
- rozmiaru, o jaki zostanie powiększona baza danych, gdy zbliży się do przekroczenia rozmiaru wejściowego,
- modelu odtwarzania bazy danych,
- grup plików i pliki,
- opcji konfiguracyjnych.

Nie będziemy szczegółowo wyjaśniać wszystkich możliwości konfiguracyjnych, ponieważ wykracza to zdecydowanie poza ramy scenariusza.

Na rysunku 24 pokazano postać okna po wybraniu zakładki Options zawierającej wiele różnych parametrów konfiguracyjnych. Przy ustalaniu wartości tych parametrów obowiązuje niepisana zasada – jeżeli nie jesteś w pełni świadomy celu, dla którego chcesz zmieniać ustawienia domyślne, to ich nie zmieniaj.



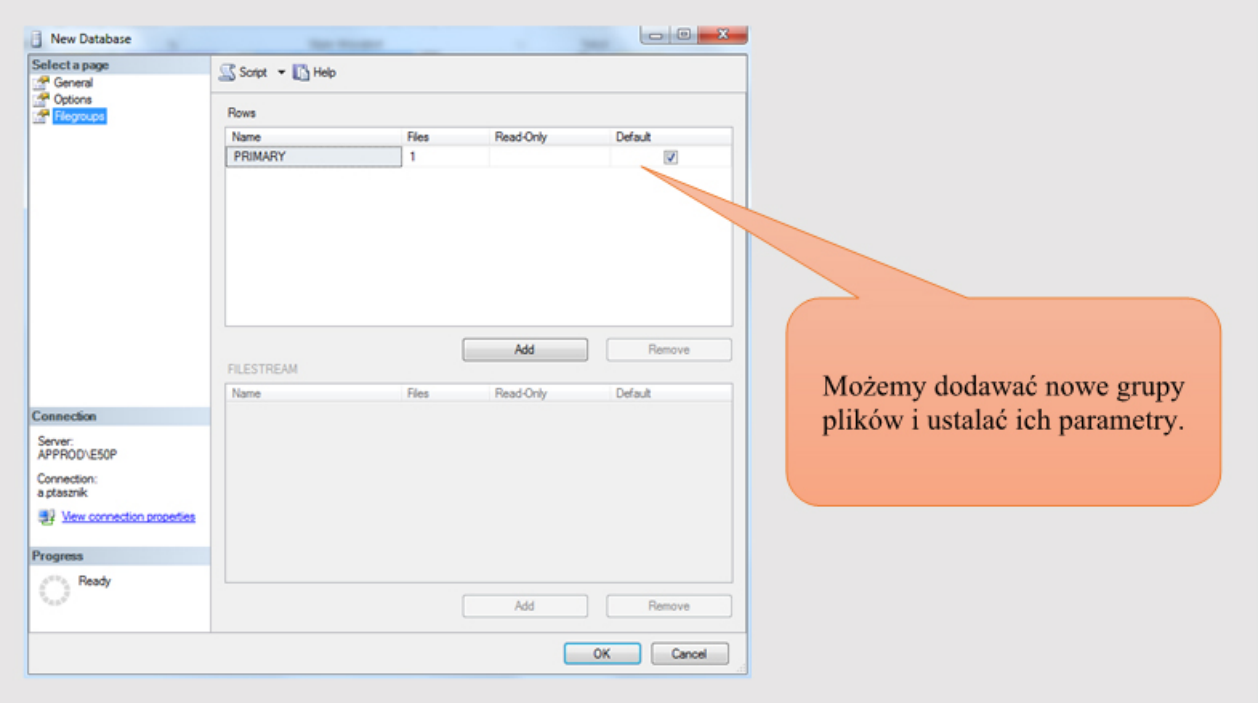
Rysunek 24. Postać okna zakładki Options.

W polu Recovery model wybieramy wartość Simple (o ile jest wybrana wartość Full). Przestawienie tej opcji jest konieczne, ponieważ wartość Full wymaga uruchomienia automatycznego wykonywania kopii dziennika transakcyjnego (jest to konieczne w systemach produkcyjnych), a nie ma większego znaczenia przy tworzeniu szkoleniowych baz danych.

Widoczne na rysunku opcje konfiguracyjne, jak wcześniej pisano, są ustawione przez producenta tak, żeby w większości zastosowań nie było potrzeby ich zmiany. Modyfikując domyślną opcję musimy zdawać sobie sprawę ze skutków dokonanej zmiany i być przekonani, że tego potrzebujemy do prawidłowego działania naszej bazy danych.

Na rysunku 24 wskazana została opcja o nazwie Encryption Enabled, dla której ustawiona jest wartość False. Interpretując pobieżnie nazwę opcji i wartość ustawionego parametru możemy uznać, że w naszej bazie danych szyfrowanie nie jest dostępne, a przestawienie parametru tej opcji na wartość True spowoduje, że będziemy mogli w naszej bazie danych wykorzystywać szyfrowanie danych wtedy, gdy będziemy tego potrzebowali. W rzeczywistości opcja ta umożliwia pełne włączenie szyfrowania wszystkich zapisów w naszej bazie danych w tle, czyli bez naszej dodatkowej ingerencji. Pozornie nie widać problemu, że nasze dane będą dodatkowo zabezpieczone odpowiednią formą ich szyfrowania, ale za wszystko trzeba płacić. Pełne szyfrowanie wszystkich zapisów wymaga wykonywania dużej liczby dodatkowych operacji, co w wielu sytuacjach może doprowadzić do niedopuszczalnego spadku wydajności systemu.

Na rysunku 25 pokazano postać okna przy wyborze zakładki Filegroup. W tym oknie możemy definiować nowe grupy plików. Konieczność planowania dodatkowych grup plików przy tworzeniu bazy danych występuje wtedy, gdy tworzymy dużą bazę danych i grupy plików będą wykorzystywane do poprawy wydajności. Ponieważ zagadnienia związane z wydajnością baz danych wykraczają zdecydowanie poza zakres naszego scenariusza, to nie będziemy tutaj więcej ich omawiać.

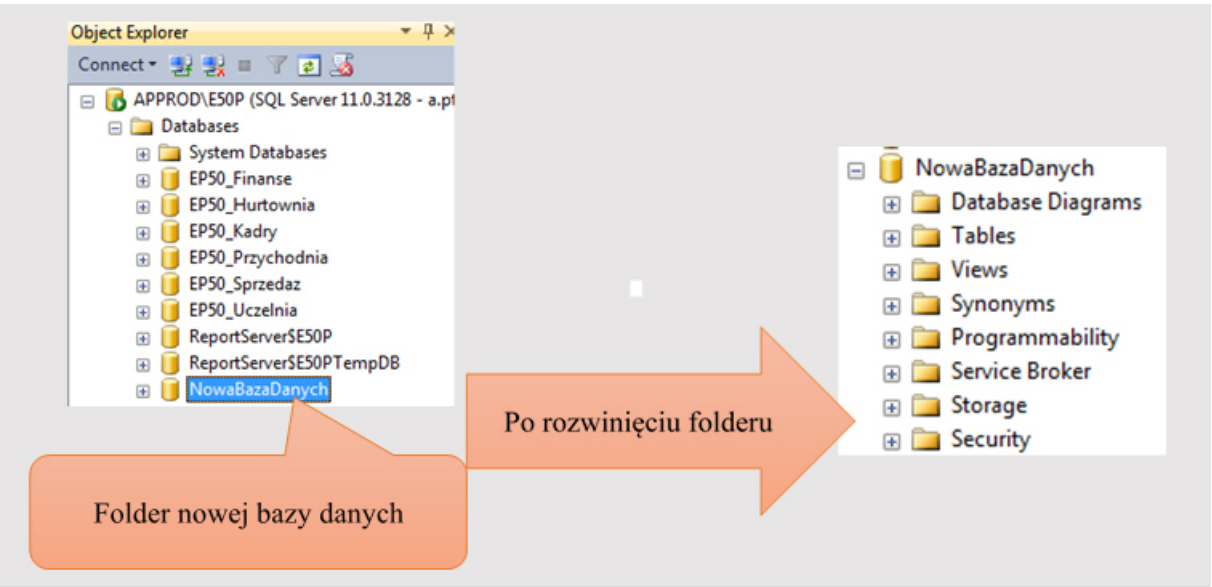


Rysunek 25. Postać okna – zakładka Filegroup.

Poznaliśmy ogólnie możliwości, jakie stwarza środowisko SQL Server 2012 w procesie tworzenia baz danych. Z jednej strony można mieć wrażenie, że jest to proces złożony wymagający perfekcyjnej znajomości technologii – i jest to prawda. Z drugiej strony dla wielu zastosowań jest to proces bardzo prosty – i to też jest stwierdzenie prawdziwe.

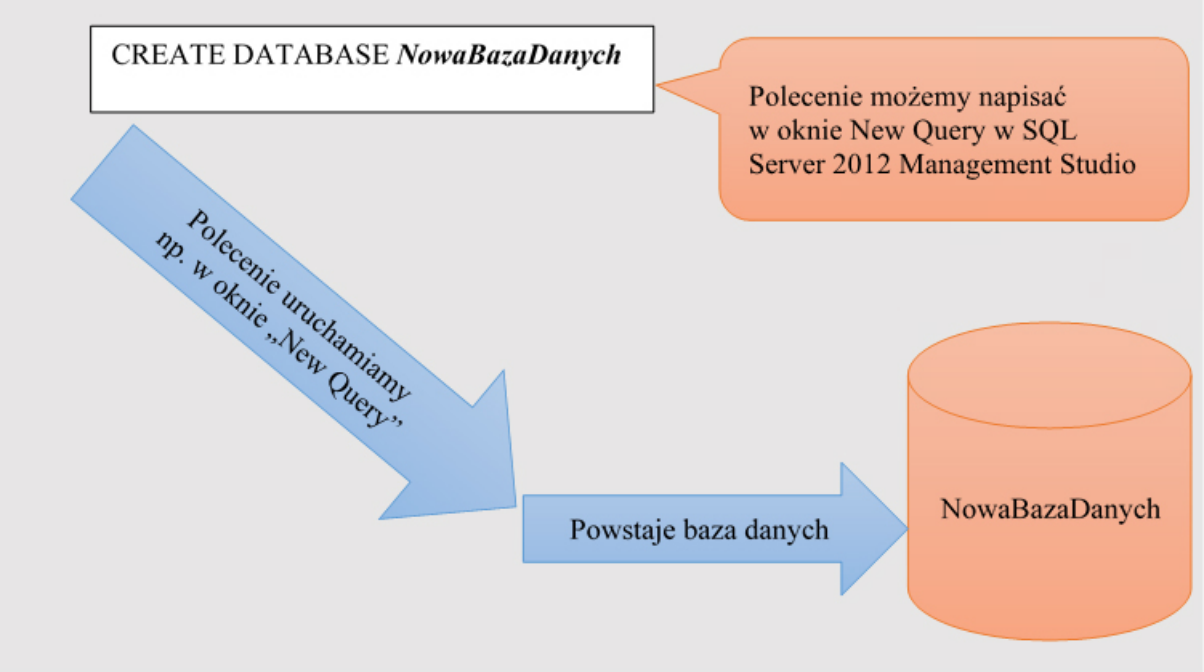
Jeżeli zakończyliśmy ustalenia związane z parametryzacją nowej bazy danych, możemy, poprzez kliknięcie przycisku OK, rozpocząć proces tworzenia bazy danych.

Po zakończeniu działania w oknie Object Explorer, jak pokazano na rysunku 26, pojawi się folder utworzonej bazy danych.



Rysunek 26. Postać okna Object Explorer po utworzeniu bazy danych.

Wspominaliśmy wcześniej, że wszystkie zadania realizowane przez SQL Server 2012 wymagają przekazania odpowiedniego polecenia w języku SQL. Rysunek 27 przedstawia postać polecenia SQL, które spowodowałoby utworzenie nowej bazy danych według domyślnych ustawień parametrów konfiguracyjnych.

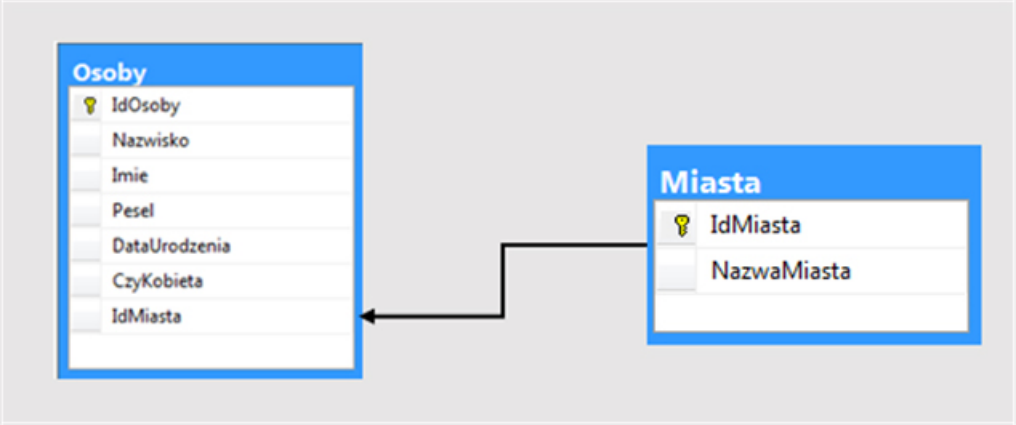


Rysunek 27. Schemat wykonania polecenia SQL.

Nowo utworzona baza danych jest kontenerem, w którym będziemy mogli rozpocząć jej organizowanie. Podstawowym zadaniem będzie utworzenie tabel, gdyż bez nich baza danych nie przedstawia żadnej wartości użytkowej. W dalszej części lekcji zapoznamy się z procesem definiowania tabel zgodnie z wcześniej przygotowanym projektem.

Tworzenie tabel w SQL Server 2012

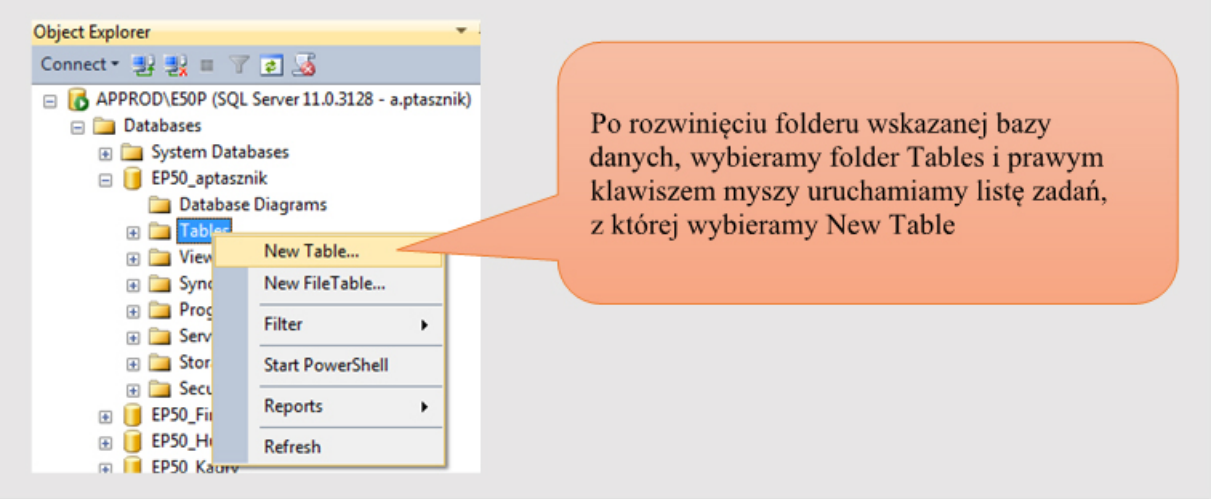
Po utworzeniu nowej bazy danych jest ona pusta, czyli nie zawiera tabel (za wyjątkiem tabel systemowych). Powiedziano wcześniej, że pierwszym krokiem przy tworzeniu baz danych jest wykonanie projektu. W celu omówienia sposobu definiowania tabel w środowisku SQL Server 2012 przedstawimy utworzenie tabel, których strukturę pokazano na rysunku 28.



Rysunek 28. Fragment projektu bazy danych.

Pokazane na rysunku 28 tabele utworzymy w bazie danych, która została wcześniej przygotowana.

Po zalogowaniu się do serwera SQL Server 2012, w oknie ObjectExplorer wybieramy nazwę bazy danych, a po rozwinięciu folderu wybieramy folder Tables, po czym prawym klawiszem myszy uruchamiamy listę zadań, z której zaznaczamy zadanie New Table. Schemat wyboru opcji New Table ukazuje rysunek 29.



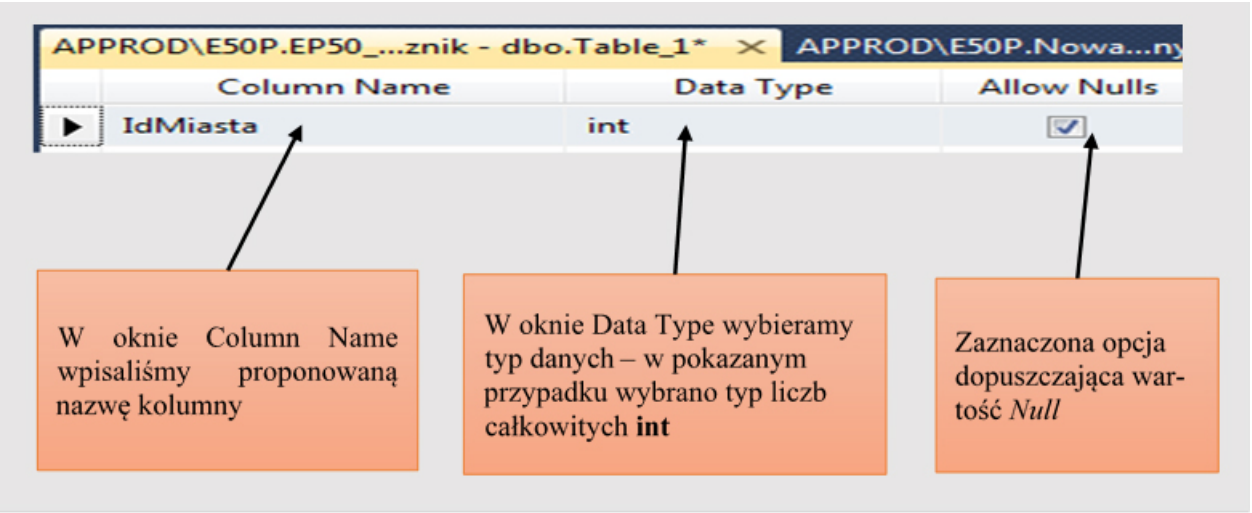
Rysunek 29. Wybór zadania New Table.

Po wybraniu opcji New Table pojawia się okno, w którym będziemy mogli definiować kolejne kolumny w tabeli. W pierwszej kolejności opiszemy proces tworzenia tabeli o nazwie `Miasta`, której strukturę pokazano na rysunku 28.

W trakcie definiowania nowej tabeli powinniśmy zrealizować następujące elementy:

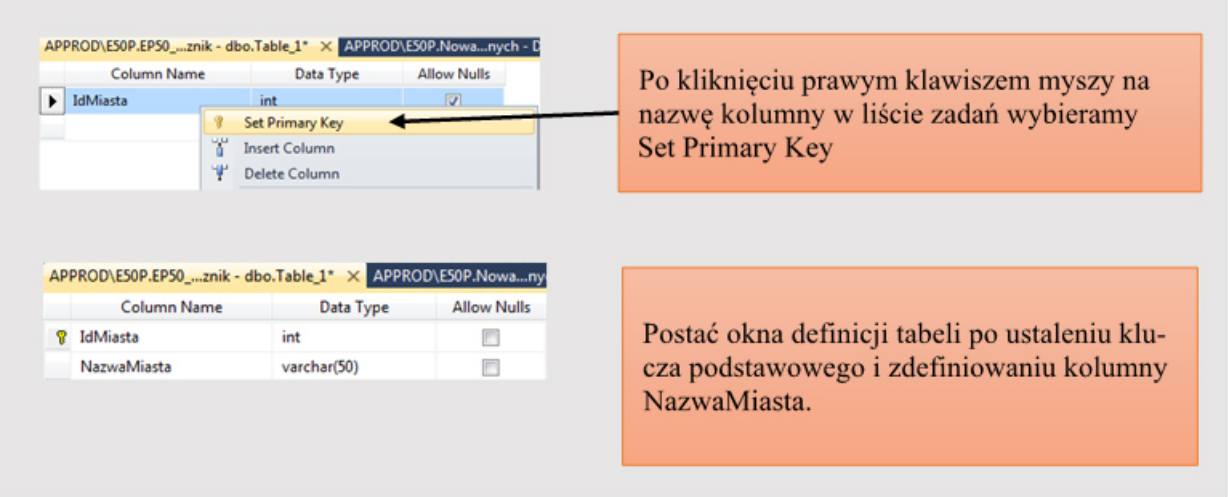
- określić kolumnę klucza podstawowego i zdefiniować dla tej kolumny mechanizm automatycznego nadawania wartości klucza,
- określić nazwy i typ danych pozostałych kolumn oraz zdecydować, czy w danej kolumnie można przechowywać wartość `NULL`,
- zakończyć proces definicji struktury tabeli i ustalić jej nazwę.

Postać okna, które zostanie otwarte po wybraniu opcji New Table widać na kolejnym rysunku.



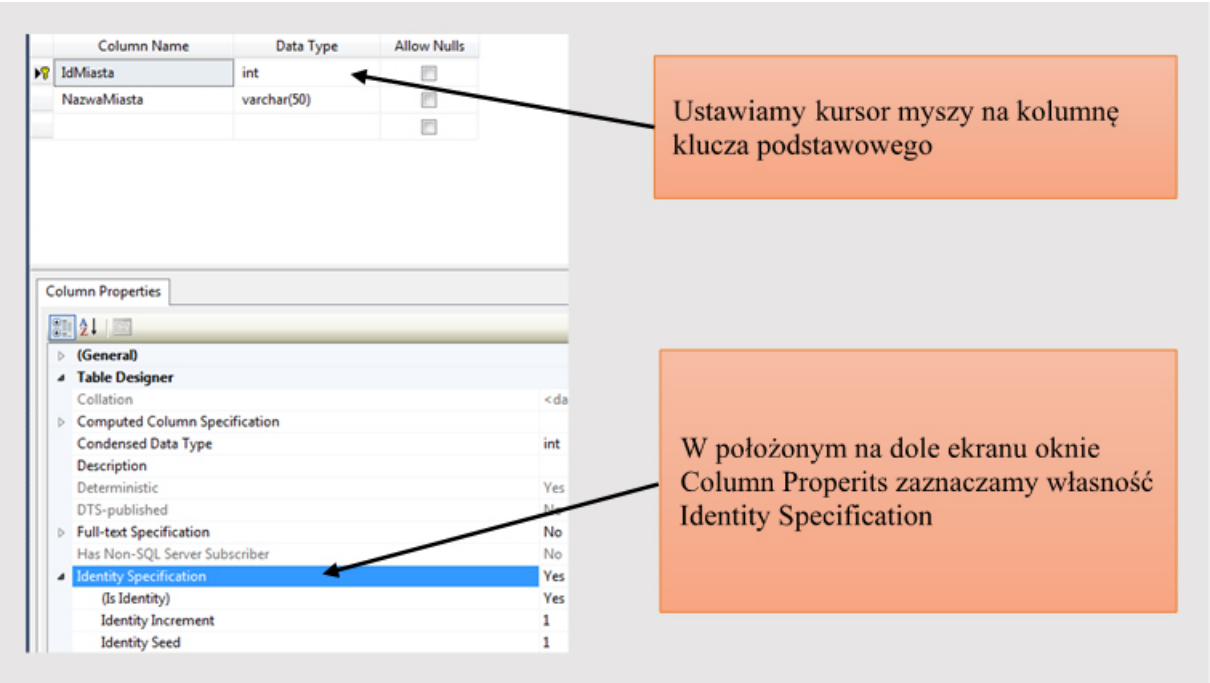
Rysunek 30. Kolejne kroki przy definiowaniu tabeli **`Miasta`**.

W kolejnym kroku zdefiniujemy kolumnę o nazwie IdMiasta jako klucz podstawowy, skutkiem czego będzie automatyczne odznaczenie opcji dopuszczającej wartość `NULL` (zgodnie z definicją, klucz podstawowy nie może przyjmować wartości `NULL`). Postać okna przy definicji klucza podstawowego pokazano na rysunku 31.



Rysunek 31. Definiowanie klucza podstawowego i pozostałych kolumn.

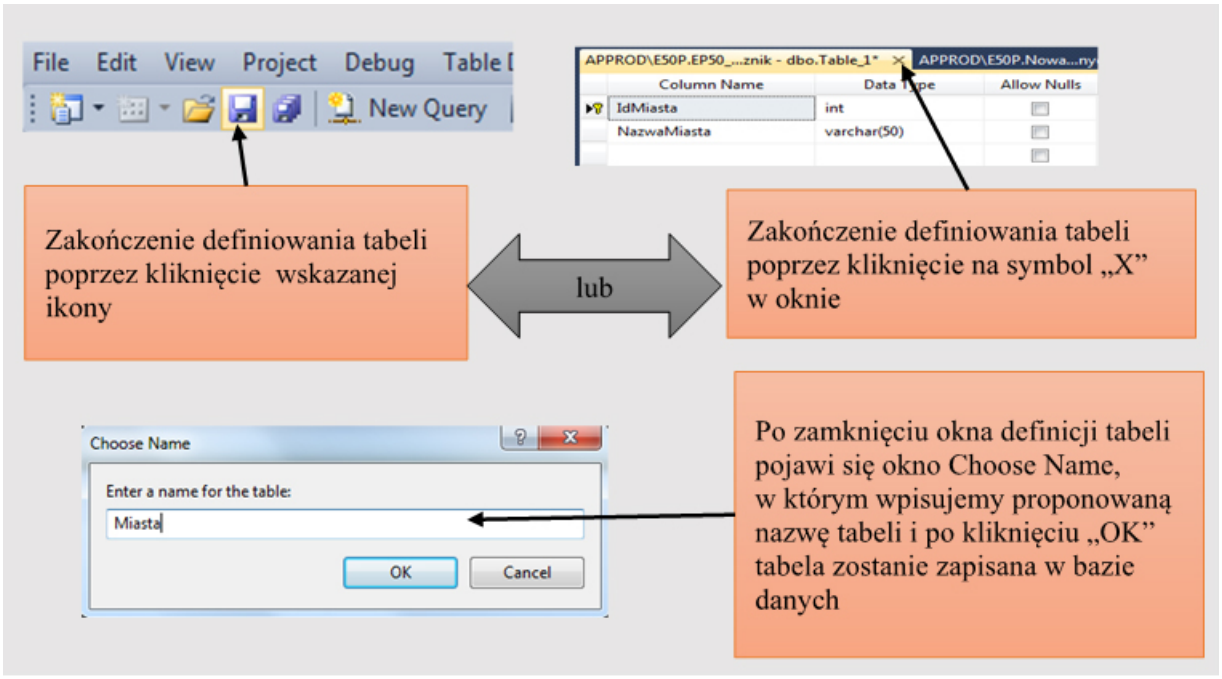
Pokazana na rysunku 30 postać definicji tabeli o nazwie `Miasta` jest już praktyczne gotowa do zapisania w bazie danych, pozostaje jednak podjęcie decyzji o sposobie ustalania wartości dla kolumny klucza podstawowego. Najczęściej w takich sytuacjach uruchamiany jest mechanizm autonumeracji. Sposób uruchomienia autonumeracji pokazano na rysunku 32.



Rysunek 32. Ustalanie mechanizmu autonumeracji.

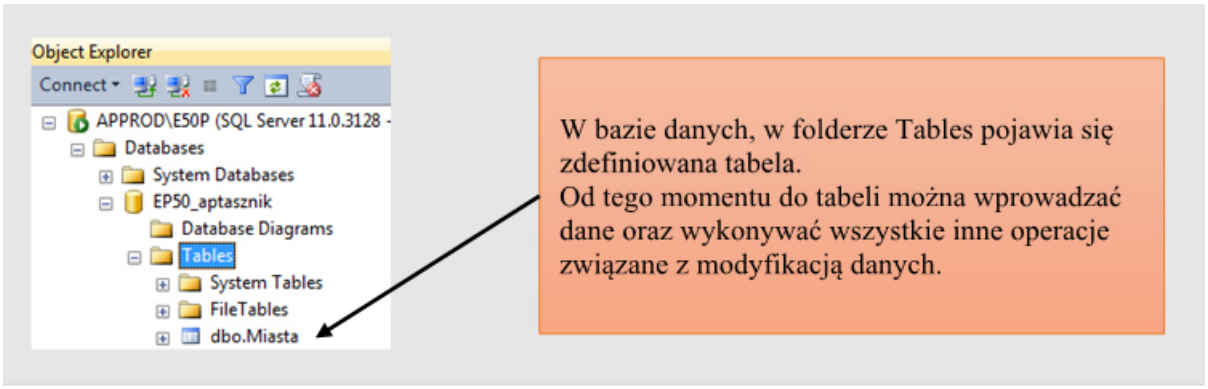
Po zaznaczeniu własności Identity Specification, jak pokazano na rysunku 32, tabela jest już gotowa do zapisania w bazie danych.

Na kolejnym rysunku widzimy metody zamknięcia okna definicji tabeli oraz sposób zapisania jej nazwy.



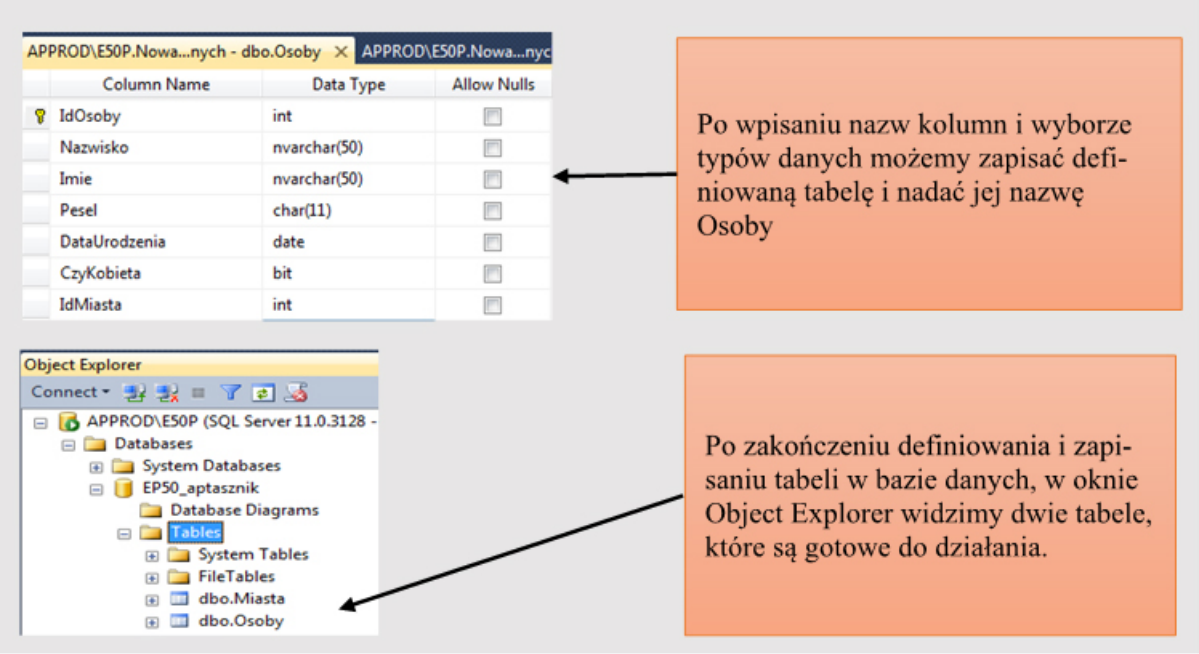
Rysunek 33. Proces zakończenia definiowania tabeli **`Miasta`**.

Zakończenie procesu definiowania tabeli spowoduje, że w bazie danych zostanie utworzona, zgodnie z definicją, tabela jak na rysunku 34.



Rysunek 34. Postać okna „Object Explorer" po zdefiniowaniu tabeli **`Miasta`**.

Proces definiowania tabeli `Osoby` jest podobny jak pokazany wcześniej, jedynym problemem jest ustalenie odpowiednich typów danych, gdyż ogólne zasady są takie same przy definicji dowolnej tabeli. Na rysunku 35 pokazano okno definicji tabeli `Osoby`.



Rysunek 35. Postać okna definiowania tabeli **`Osoby`**.

Wyjaśnienia wymaga nazwa kolumny CzyKobieta. Taki wybór nazwy kolumny związany jest z typem danych, który dla tej kolumny został wybrany, czyli typ bit. Ponieważ typ bit jest typem danych, który umożliwia zapisanie tylko dwóch wartości (prawda lub fałsz), to dobrym pomysłem jest, żeby nazwa kolumny miała formę pytania – łatwo wtedy interpretować te dane. Wyobraźmy sobie, że kolumna nosi nazwę Płeć (bo tak naprawdę określamy w tej kolumnie płeć osoby) i w takiej sytuacji zawsze potrzebowalibyśmy informacji (a pamięć ludzka jest ulotna), jaką zasadę przyjęto, czyli czy wartość prawda w kolumnie „Płeć" określa mężczyznę czy kobietę.

Podobnie jak w przypadku definiowania tabeli `Miasta`, powinniśmy teraz zamknąć okno i ustalić nazwę tabeli. Po zapisaniu w bazie danych dysponujemy dwoma tabelami gotowymi do działania.

Pozornie wydawać by się mogło, że zdefiniowane tabele są w pełni gotowe do działania i to jest prawdą, jednak w ramach kolejnego modułu tematycznego poświęconego spójności i integralności danych zobaczymy, że czeka nas jeszcze trochę pracy, żeby gromadzić poprawne dane.

Podstawa programowa

Etap edukacyjny: IV, przedmiot: informatyka (poziom podstawowy)

Cele kształcenia – wymagania ogólne

I. Wyszukiwanie, gromadzenie i przetwarzanie informacji z różnych źródeł; opracowywanie za pomocą komputera: rysunków, tekstów, danych liczbowych, motywów, animacji, prezentacji multimedialnych.

Treści nauczania – wymagania szczegółowe

4. Opracowywanie informacji za pomocą komputera, w tym: rysunków, tekstów, danych liczbowych, animacji, prezentacji multimedialnych i filmów. Uczeń:
6. tworzy bazę danych, posługuje się formularzami, porządkuje dane, wyszukuje informacje, stosując filtrowanie;
7. wykonuje podstawowe operacje modyfikowania i wyszukiwania informacji na relacyjnej bazie danych.

Cel

Podstawowym celem lekcji jest utworzenie bazy danych i zdefiniowanie tabel, których postać została zaproponowana przez uczniów.

Słowa kluczowe

SQL Server ManagementStudio, typy danych, klucz podstawowy, klucz obcy

Co przygotować

- Przed przystąpieniem uczniów do samodzielnej pracy należy wspólnie pokazać proces tworzenia bazy danych i definiowania przykładowej tabeli.

Przebieg zajęć

Wprowadzenie (10 minut)

W trakcie wprowadzenia należy pokazać uczniom sposób logowania do środowiska SQL Server oraz tworzenia nowej bazy danych i tabeli

Praca w zespołach (30 minut)

Grupy uczniów tworzą bazę danych oraz definiują tabele, które wcześniej zaprojektowali dla wybranej dziedziny problemu.

Dyskusja podsumowująca

W ramach dyskusji należy ocenić zrealizowane projekty i omówić ewentualne błędy.

Sprawdzenie wiedzy

W celu sprawdzenia wiedzy można wykorzystać zamieszczony w materiałach scenariusza test 2.

Ocenianie

Ocena uczniów na podstawie testu oraz oceny zrealizowanego projektu.

Dostępne pliki

1. Test 2
2. Film 3 – Definiowanie tabel

LEKCJA NR 3

TEMAT: Definiowanie ograniczeń

Streszczenie

Wprowadzenie do problematyki integralności danych

Pierwszym skojarzeniem związanym z relacyjną bazą danych jest zwykle tabela lub kilka tabel, które opisują pewną dziedzinę Niestety, w tym miejscu kończy się wiedza części programistów i w tworzonych przez nich aplikacjach baza danych jest wykorzystywana jedynie jako prymitywna składnica danych. Wszelkie operacje weryfikacji poprawności danych, zapewnienia im spójności oraz samo przetwarzanie danych odbywa się w ramach logiki aplikacji.

Co więc można zrobić, żeby zadbać w większym stopniu o cenne dane, które chcemy gromadzić w bazie? Przede wszystkim poznać nieco bliżej możliwości oferowane przez relacyjne bazy danych w zakresie zapewnienia spójności przechowywanym danym. W ramach lekcji postaramy się przybliżyć kilka takich mechanizmów, a także zaprezentować dodatkowe informacje, które pozwolą spojrzeć na bazę danych, jako na twór, który można w bardzo szerokim zakresie kształtować, wzbogacać jego funkcjonalność.

Po utworzeniu bazy danych i zdefiniowaniu odpowiednich tabel może się wydawać, że baza danych jest w pełni gotowa do realizacji postawionych przed nią zadań. Problem tkwi w tym, że jednym z najważniejszych zadań stojących przed bazami jest ich wiarygodność, a tę można zapewnić tylko poprzez wyeliminowanie, w możliwie największym stopniu, błędów pojawiających się w trakcie zapisywania i modyfikacji danych w bazie. Stu procentowa poprawność danych jest jednak praktycznie nieosiągalna, ale koniecznością jest zapewnienie maksymalnego poziomu poprawności danych.

Systemy Zarządzania Relacyjnymi Bazami Danych dostarczają mechanizmów służących do zapewnienia spójności i integralności danych, czyli mówiąc innymi słowami, zapewnienia logicznej poprawności danych zapisanych w bazie. Podstawowe mechanizmy realizujące te zadania to:

- deklaracja typu,
- deklaracja dopuszczalności wartości `NULL`,
- deklaracja wartości domyślnych,
- definicje kluczy,
- reguły poprawności dla kolumny,
- reguły poprawności dla wiersza,
- reguły integralności referencyjnej.

W dalszej części lekcji zostaną zdefiniowane ograniczenia dla utworzonych tabel.

Typy danych

Deklaracja typu danych, w przeciwieństwie do pozostałych mechanizmów zapewnienia poprawności danych, musi być realizowana obowiązkowo już na etapie tworzenia tabeli.

Deklaracja typu danych: Nie istnieje możliwość utworzenia tabeli bez określenia typu danych dla każdej kolumny.

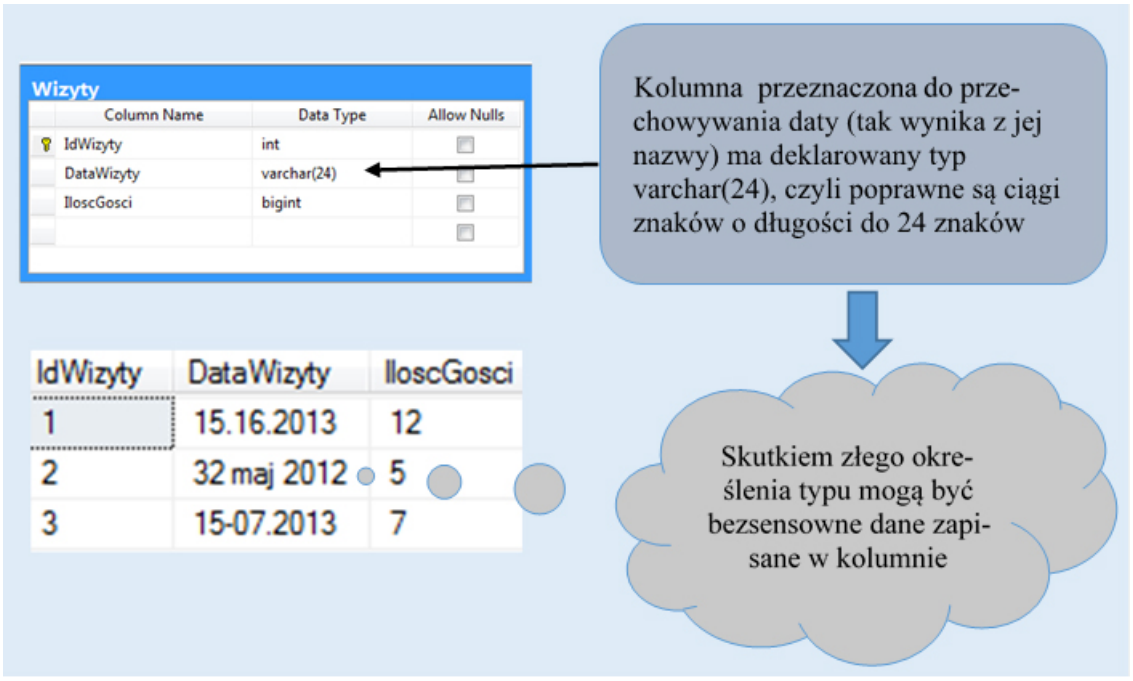
W środowisku SQL Server 2012 dostępnych jest wiele wbudowanych typów danych, które można podzielić na następujące kategorie:

- typy liczb całkowitych,
- typy liczb rzeczywistych,
- typy znakowe,
- typy daty i czasu,
- typy różne,
- typy nierelacyjne.

Pozornie decyzja o wyborze typu danych dla kolumny wydaje się zadaniem prostym, ale złe wybory mogą mieć poważne konsekwencje w trakcie eksploatacji bazy danych. Podstawowe zagrożenia przy deklarowaniu typu danych to:

- wybór niewłaściwego typu danych,
- wybór nadmiarowego typu danych.

Przykładem niewłaściwego typu danych może być sytuacja, gdy dla kolumny, w której planujemy przechowywać daty wybrany został typ znakowy, co może spowodować zapisywanie w bazie danych błędnych wartości. Będą one poprawne z punktu widzenia deklarowanego typu danych, czyli będą odpowiednim ciągiem znaków, ale ten ciąg znaków nie będzie reprezentował poprawnej daty. Omawianą sytuację pokazano na rysunku 36.



Rysunek 36. Przykład niewłaściwego wyboru typu danych.

W przykładzie ukazanym na rysunku 36 zilustrowano również sytuację nadmiarowego typu danych. Dla kolumny o nazwie IloscGosci zadeklarowano typ danych o nazwie bigint. Zakładamy, że wartości w kolumnie o nazwie IloscGosci będą przyjmowały wartości z zakresu od 0 do 100. Z jednej strony wybór typu jest poprawny, ponieważ w kolumnie będą umieszczone liczby całkowite, a typ bigint to liczba całkowita z zakresu [-9223372036854775808 – 922337203685477580], do przechowywania której wykorzystuje się osiem bajtów. Dla założonych tu wartości można było wykorzystać typ danych tinyint, który jest liczbą całkowitą z zakresu [0 – 255], przechowywaną z wykorzystaniem jednego bajtu.

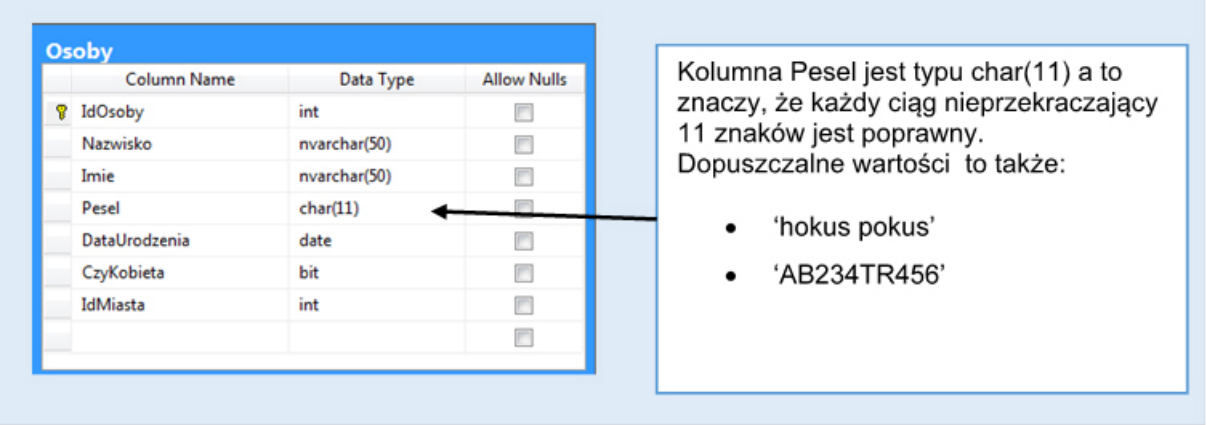
Z tego przykładu wynika, że do zapisania wartości w kolumnie o nazwie IloscGosci, należy wykorzystać 8 bajtów, a można byłoby wykorzystać 1 bajt. Pozostaje do wyjaśnienia, czy 7 nadmiarowych bajtów stanowi poważny problem dla bazy danych. Patrząc na jeden wiersz tabeli można ten problem zlekceważyć,

gdyż 7 bajtów to naprawdę niewielki obszar pamięci, który będzie zbędnie wykorzystany przy przechowaniu danych. Jednak w tabelach relacyjnych mogą być zapisane miliony wierszy i wtedy niepotrzebnie wykorzystywana do zapisania danych pamięć stanowi już bardzo znaczący obszar.

Deklaracja typu jest pierwszym krokiem na drodze zapewnienia poprawności przechowywanych danych, który musimy zrealizować już na etapie definiowania struktury tabeli.

Reguły poprawności dla wiersza

Problemy związane z regułami poprawności przeanalizujemy na przykładzie kolumny o nazwie Pesel w pokazanej na rysunku 37 przykładowej tabeli.



Rysunek 37. Struktura tabeli **Osoby**.

Załóżmy, że dla kolumny Pesel został zdefiniowany typ danych char(11). Wydaje się, że jest to definicja poprawna, ale rodzi się wiele problemów związanych z zapewnieniem poprawności danych zapisywanych w tej kolumnie, ponieważ musimy wymusić, żeby każda wartość zapisana w tej kolumnie była poprawnym numerem pesel.

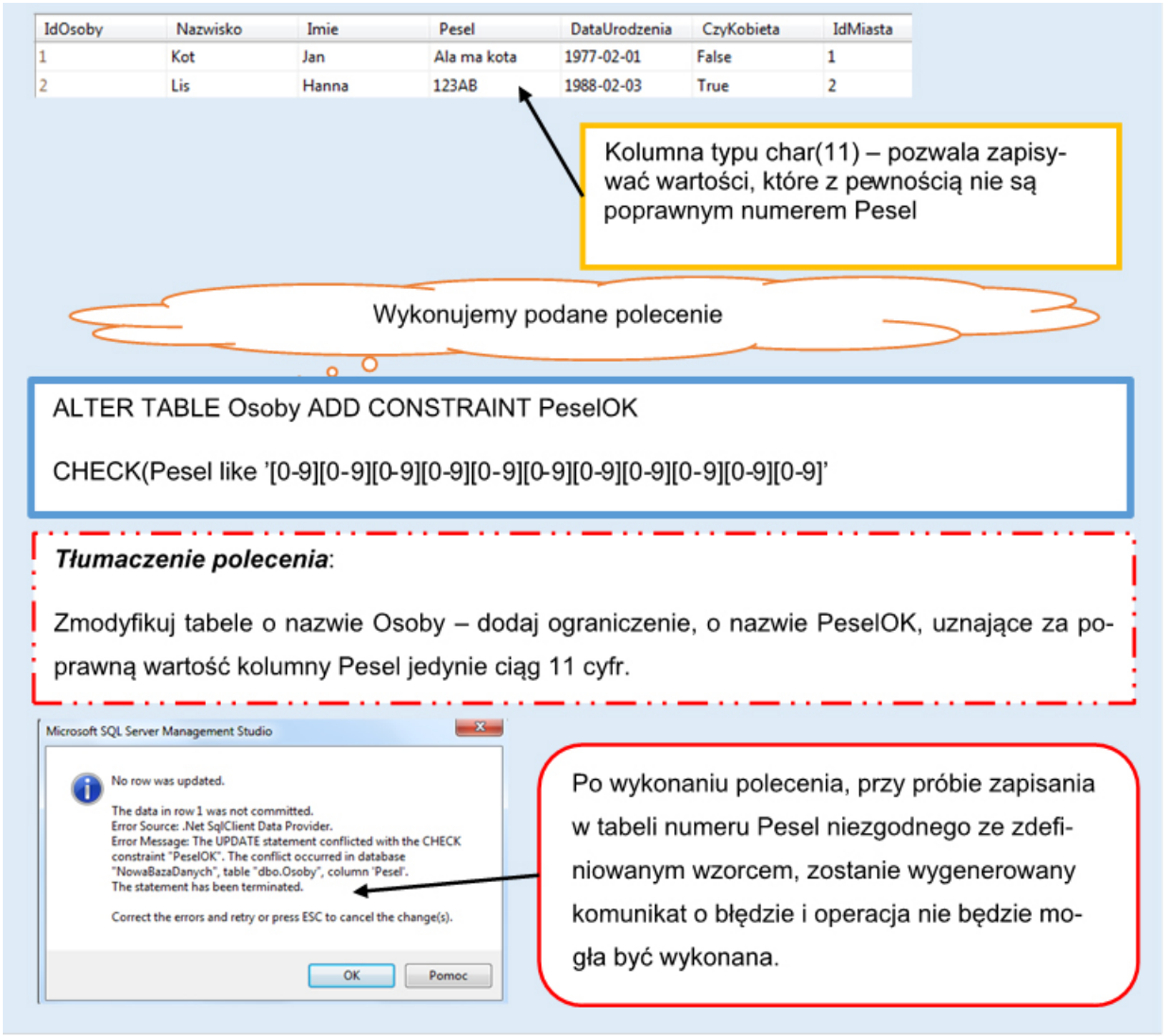
Z punktu widzenia deklaracji typu char(11), poprawnymi wartościami są wszystkie ciągi znakowe o długości nieprzekraczającej 11 znaków i w tym momencie widzimy, jak daleko jeszcze do pełnej poprawności. Gdybyśmy pozostali na takim zdefiniowaniu tej kolumny, to za poprawne dane mogłyby uchodzić nawet tak bezsensowne dane jak:

- Ala ma kota,
- Aw44,
- brak Pesel.

Każdy z tych przykładowych ciągów znakowych jest poprawny, ponieważ nie przekracza 11 znaków. W celu zapewnienia poprawności danych powinniśmy wymusić, żeby zapisywane numery Pesel składały się dokładnie z 11 cyfr. W celu realizacji tego zadania można skorzystać z mechanizmu definiowania ograniczeń. Ograniczenie dziedziny wartości sprowadza się do zdefiniowania wyrażenia logicznego, które, jeśli jest spełnione, uznaje dane za poprawne. W naszym przypadku takie wyrażenie mogłoby mieć następującą postać:

```
Pesel LIKE ' [0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'
```

Pokazane powyżej wyrażenie logiczne korzysta z mechanizmu wyrażeń regularnych, które służą do określenia budowy poprawnych ciągów znakowych. W naszym wypadku skorzystaliśmy z wyrażenia [0..9], które określa, że w tym miejscu powinien wystąpić znak z zakresu od zero do dziewięć, czyli dowolna cyfra. Ponieważ powtórzyliśmy omawiany symbol 11 razy to znaczy, że zgodny z tym wyrażeniem ciąg znaków musi składać się z dokładnie 11 cyfr. Na rysunku 38 pokazano proces zdefiniowania ograniczenia CHECK dla kolumny za pomocą polecenia w języku SQL.

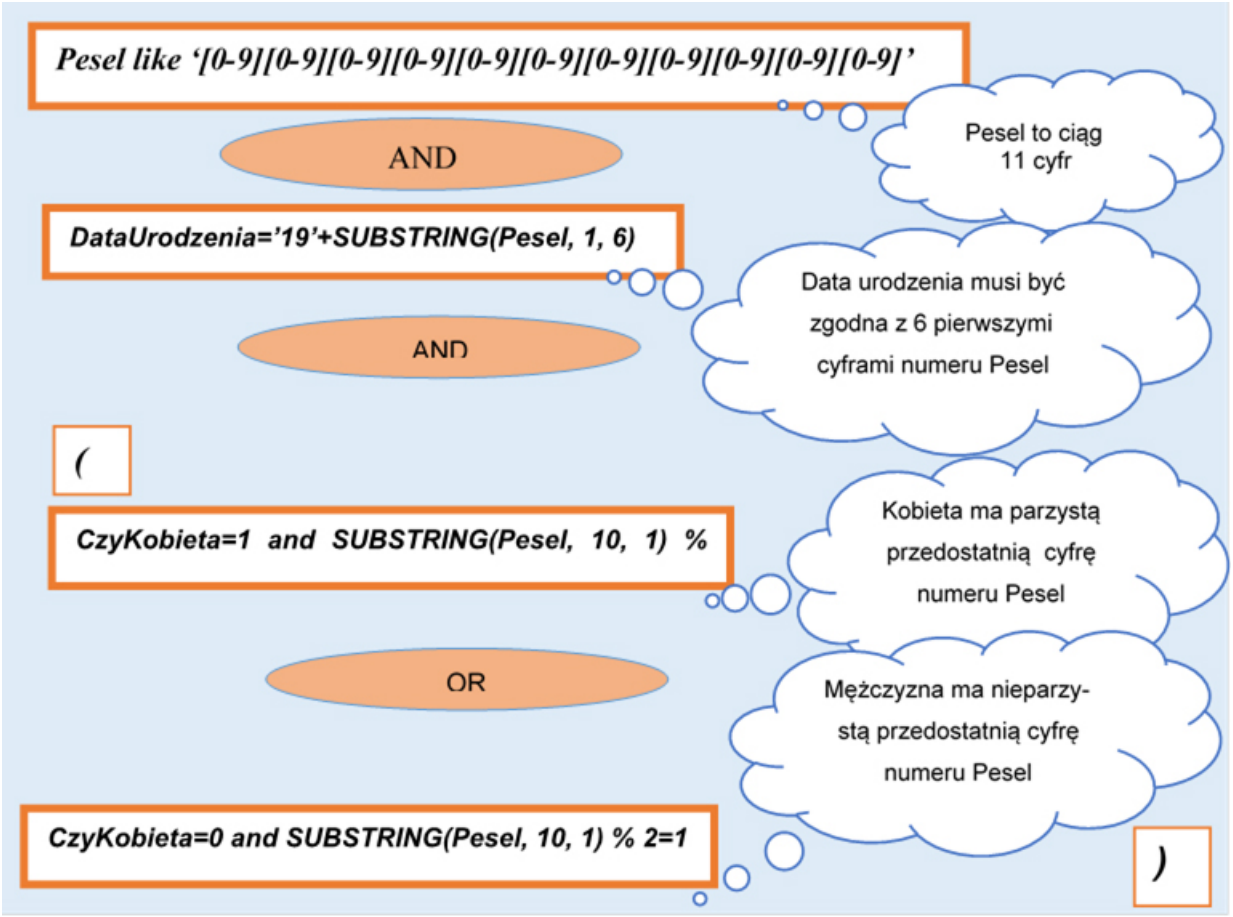


Rysunek 38. Zdefiniowanie ograniczenia CHECK za pomocą polecenia w języku SQL.

Pomimo zdefiniowania reguły, zgodnie z którą numer Pesel musi być zapisany jako 11 cyfr, nie zagwarantowaliśmy jeszcze poprawności danych zapisywanych w tabeli `Osoby`, ponieważ w jednym wierszu obok numeru Pesel zapisana jest data urodzenia, a także informacja o płci. Zgodnie z zasadami obowiązującymi w rejestrze Pesel w poprawnym numerze sześć pierwszych cyfr odwzoruje datę urodzenia danej osoby, a przedostatnia – płeć (cyfra parzysta to kobiety, nieparzysta to mężczyźni). Z powyższych ustaleń wynika, że zagwarantowanie jedenastocyfrowych numerów w kolumnie Pesel nie jest wystarczające dla zapewnienia poprawności danych w obrębie całego wiersza.

Rozwiązanie problemu ograniczenia CHECK dla całego wiersza, zasadniczo nie różni się od ograniczenia CHECK dla pojedynczej kolumny, ponieważ w obu przypadkach należy napisać wyrażenie logiczne, które będzie warunkiem uznania, że zapisywane dane są poprawne. Na rysunku 39 omówiono budowę wyra-

żenia logicznego dla pełnego problemu poprawności numeru Pesel. Wyrażenie logiczne może składać się z wielu pojedynczych warunków połączonych operatorami logicznymi AND (koniunkcja) lub OR (alternatywa).



Rysunek 39. Budowa wyrażenia określającego poprawność numeru Pesel.

Budowa wyrażenia logicznego dla omawianego problemu może wydać się dość skomplikowana szczególnie w początkowym okresie nauki, ale w miarę czasu i zdobywania dodatkowej wiedzy wyrażenie to okaże się całkiem proste.

Ostateczna postać polecenia definiującego ograniczenie CHECK dla problemu poprawności numeru Pesel i synchronizacji z danymi zapisanymi w kolumnach DataUrodzenia i CzyKobieta pokazana została na rysunku 40.

```
ALTER TABLE Osoby ADD CONSTRAINT PeselOK CHECK
(
  Pesel like '[0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'
 AND
  DataUrodzenia='19'+SUBSTRING(Pesel, 1, 6)

AND

        (

            CzyKobieta=1 and SUBSTRING(Pesel, 10, 1) % 2=0

        OR

            CzyKobieta=0 and SUBSTRING(Pesel, 10, 1) % 2=1

        )


)
```

Rysunek 40. Definicja ograniczenia CHECK.

Wykonanie polecenia pokazanego na rysunku 40 zagwarantuje, że wszystkie operacje modyfikacji danych w tabeli `Osoby` muszą być zgodne z tym wyrażeniem.

Klucz unikalny

Poza kluczem podstawowym w tabeli mogą istnieć kolumny, których wartości dla każdego wiersza w tabeli są unikalne. Możemy zdefiniować ograniczenie klucza unikalnego dla takich kolumn.

Na rysunku 41 pokazano polecenia SQL definiujące klucz unikalny dla kolumny Pesel w tabeli `Osoby` oraz dla kolumny Nazwa w tabeli `Miasta`.

```
ALTER TABLE   Osoby
ADD CONSTRAINT  UPesel UNIQUE(Pesel)


ALTER TABLE   Miasta
ADD CONSTRAINT  UNazwa  UNIQUE(Nazwa)
```

Rysunek 41. Definicja kluczy unikalnych.

Klucz obcy

Dla przykładowych tabel pokazanych na rysunku 28, należy zdefiniować ograniczenie klucza obcego, którego celem jest zapewnienie powiązania kolumny IdMiasta w tabeli `Osoby` z odpowiednim wierszem tabeli `Miasta`. Na rysunku 42 pokazano polecenie SQL definiujące klucz obcy.

```
ALTER TABLE Osoby
ADD CONSTRAINT FKIdMiasta FOREIGN KEY(IdMiasta)
REFERENCES Miasta (IdMiasta)
```

Rysunek 42. Definicja ograniczenia klucza obcego.

Podstawa programowa

Etap edukacyjny: IV, przedmiot: informatyka (poziom podstawowy)

Cele kształcenia – wymagania ogólne

I. Wyszukiwanie, gromadzenie i przetwarzanie informacji z różnych źródeł; opracowywanie za pomocą komputera: rysunków, tekstów, danych liczbowych, motywów, animacji, prezentacji multimedialnych.

Treści nauczania – wymagania szczegółowe

4. Opracowywanie informacji za pomocą komputera, w tym: rysunków, tekstów, danych liczbowych, animacji, prezentacji multimedialnych i filmów. Uczeń:
   6) tworzy bazę danych, posługuje się formularzami, porządkuje dane, wyszukuje informacje, stosując filtrowanie;
   7) wykonuje podstawowe operacje modyfikowania i wyszukiwania informacji na relacyjnej bazie danych.

Cel

Podstawowym celem lekcji jest wyjaśnienie istoty ograniczeń, które definiujemy w bazie danych w celu zapewnienia integralności danych.

Słowa kluczowe

relacyjny model danych, integralność referencyjna, ograniczenia, klucz unikalny, klucz obcy

Co przygotować

- Podstawą realizacji lekcji są tabele zdefiniowane w bazie danych, których strukturę pokazano na rysunku 27.

Przebieg zajęć

Wprowadzenie (30 minut)

W trakcie wprowadzenia należy omówić cel definiowania ograniczeń w bazie danych.

Praca w zespołach (30 minut)

Uczniowie w grupach definiują przykładowe ograniczenie, którego celem jest zapewnienie poprawności danych (pesel, data urodzenia, płeć).

Wprowadzanie przykładowych danych (10 minut)

Uczniowie, po zdefiniowaniu ograniczenia, wprowadzają przykładowe dane. W przypadku próby zapisania błędnych danych obserwując działanie zdefiniowanego ograniczenia.

Sprawdzenie wiedzy

Sprawdzenie wiedzy można zrealizować wykorzystując załączony test 3.

Dostępne pliki

1. Test 3
2. Filmy instruktażowe

   Film 4 – Definiowanie ograniczeń (cz. 1)

   Film 5 – Definiowanie ograniczeń (cz. 2)

Uwagi do realizacji scenariusza z wykorzystaniem programu Access

W scenariuszu lekcji „Od chaosu do bazy danych" opisano proces tworzenia bazy danych, tabel oraz definiowania ograniczeń (constraint) z wykorzystaniem programu MS SQL Server 2012 Express Edition. Ponieważ w wielu szkołach wykorzystuje się oprogramowanie Access, można zrealizować ten scenariusz na bazie tego programu, bez konieczności instalowania systemu MS SQL Server 2012 Express Edition. W ramach scenariusza realizowane są trzy podstawowe zadania:

- tworzenie bazy danych,
- tworzenie tabel,
- definiowanie ograniczeń.

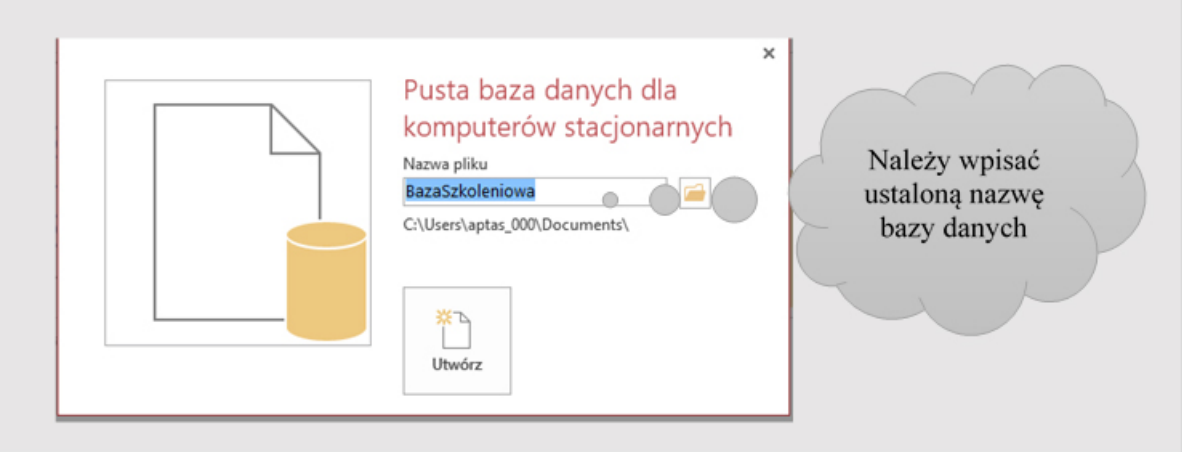
Przedstawione zostaną uwagi do realizacji tych zadań w przypadku realizowania scenariusza z wykorzystaniem programu Access 2013.

1. Tworzenie bazy danych

Proces tworzenia bazy danych jest prostszy niż w przypadku korzystania z MS SQL Server 2012. W celu utworzenia bazy danych, po uruchomieniu programu Access 2013, należy wybrać odpowiedni szablon projektu.

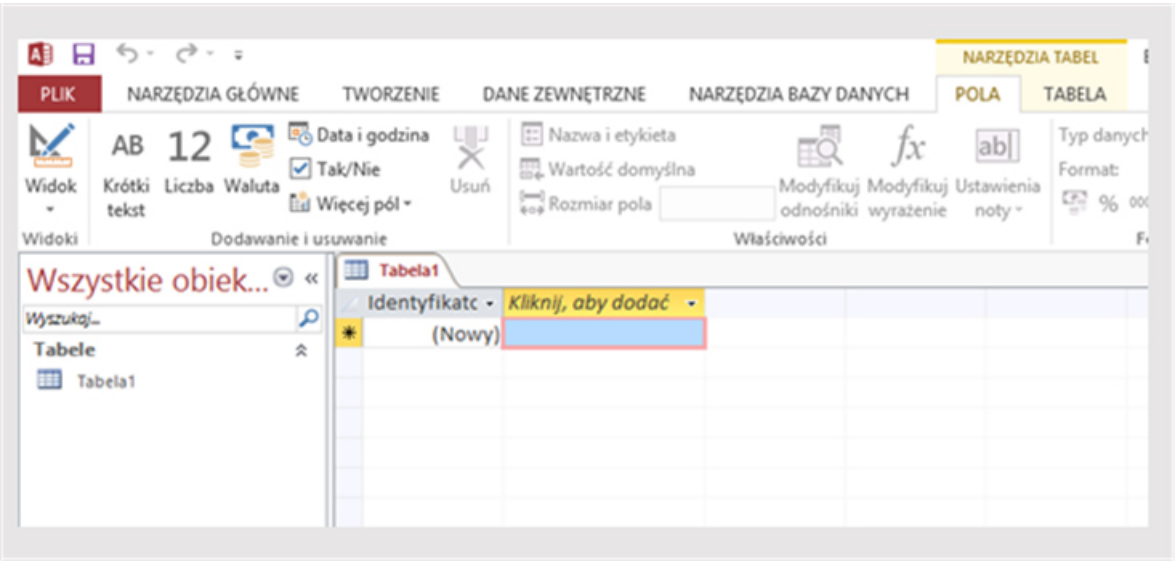
Rysunek 1. Tworzenie bazy danych w środowisku Access 2013.

Po wyborze zadania, jak pokazano na rysunku 1, otwarte zostanie okno (rysunek 2), w którym należy ustalić nazwę tworzonej bazy danych.



Rysunek 2. Ustalenie nazwy tworzonej bazy danych.

Po wpisaniu nazwy bazy danych należy kliknąć przycisk „Utwórz", zostanie uruchomiony podstawowy interfejs programu Access i możemy rozpocząć projektowanie tabel w tworzonej bazie danych. Na rysunku 3 przedstawiono postać interfejsu programu Access po utworzeniu nowej bazy danych.

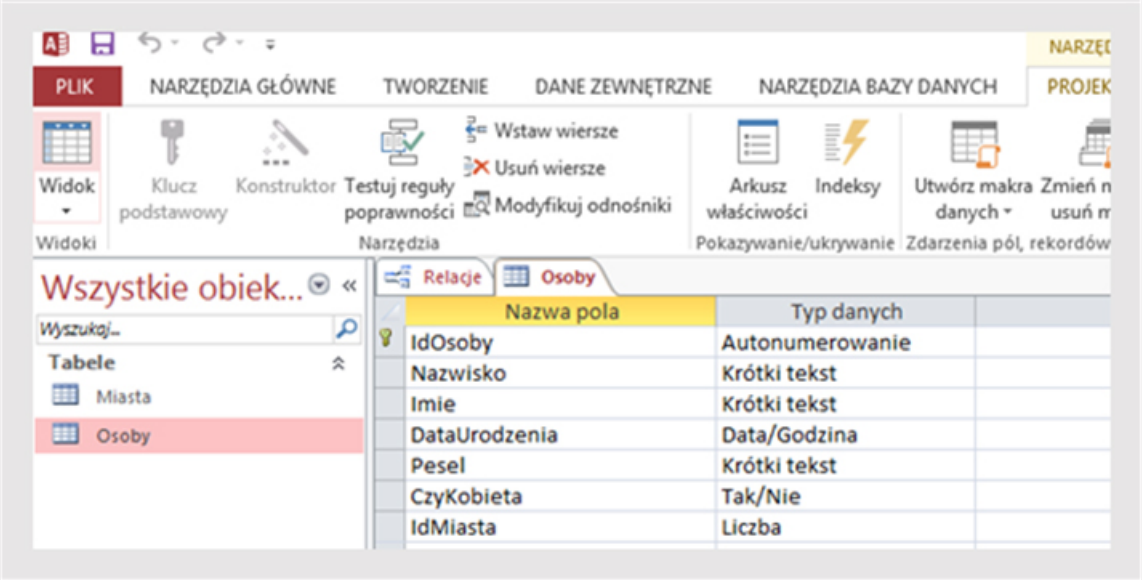


Rysunek 3. Postać interfejsu programu Access po utworzeniu nowej bazy danych.

W dalszej części prac przystępujemy do tworzenia tabel w bazie danych.

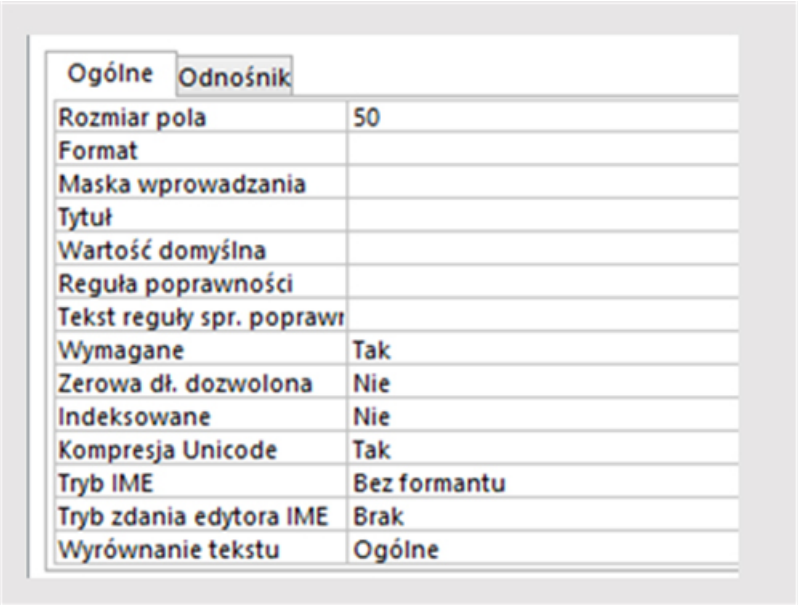
2. Tworzenie tabel

Proces tworzenia tabel przeprowadzamy w widoku projektu tabeli, jak pokazano na rysunku 4.



Rysunek 4. Widok projektu tabeli.

W kolumnie „Nazwa pola" wpisujemy nazwy kolumn projektowanej tabeli, a w kolumnie „Typ danych" wybieramy odpowiadający danej kolumnie typ danych. Dla każdej definiowanej kolumny można ustalić dodatkowe właściwości z listy dostępnej poniżej obszaru definiowania kolumn. Postać okna właściwości pokazano na rysunku 5.



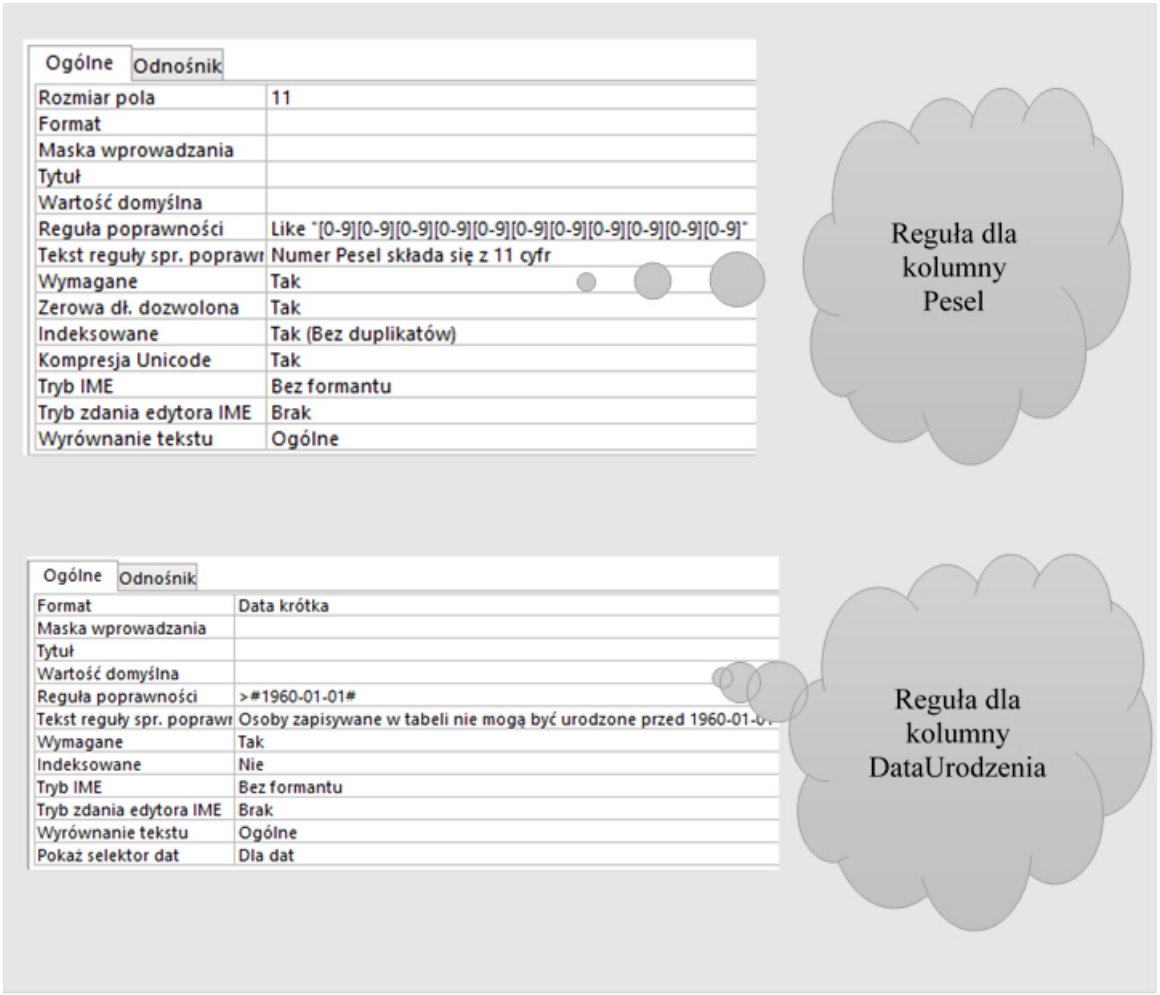
Rysunek 5. Okno właściwości kolumny.

Korzystając z widoku projektu tabeli powtarzamy proces definiowania dla każdej tabeli, która powinna znaleźć się w projektowanej bazie danych.

3. Definiowanie ograniczeń

Definiowania ograniczeń dla kolumn i wierszy w programie Access nie można zrealizować za pomocą odpowiednich poleceń języka SQL, jak pokazano w podstawowym materiale scenariusza. Podstawowe ograniczenia dla kolumn możemy zdefiniować w widoku projektu tabeli w oknie właściwości kolumny.

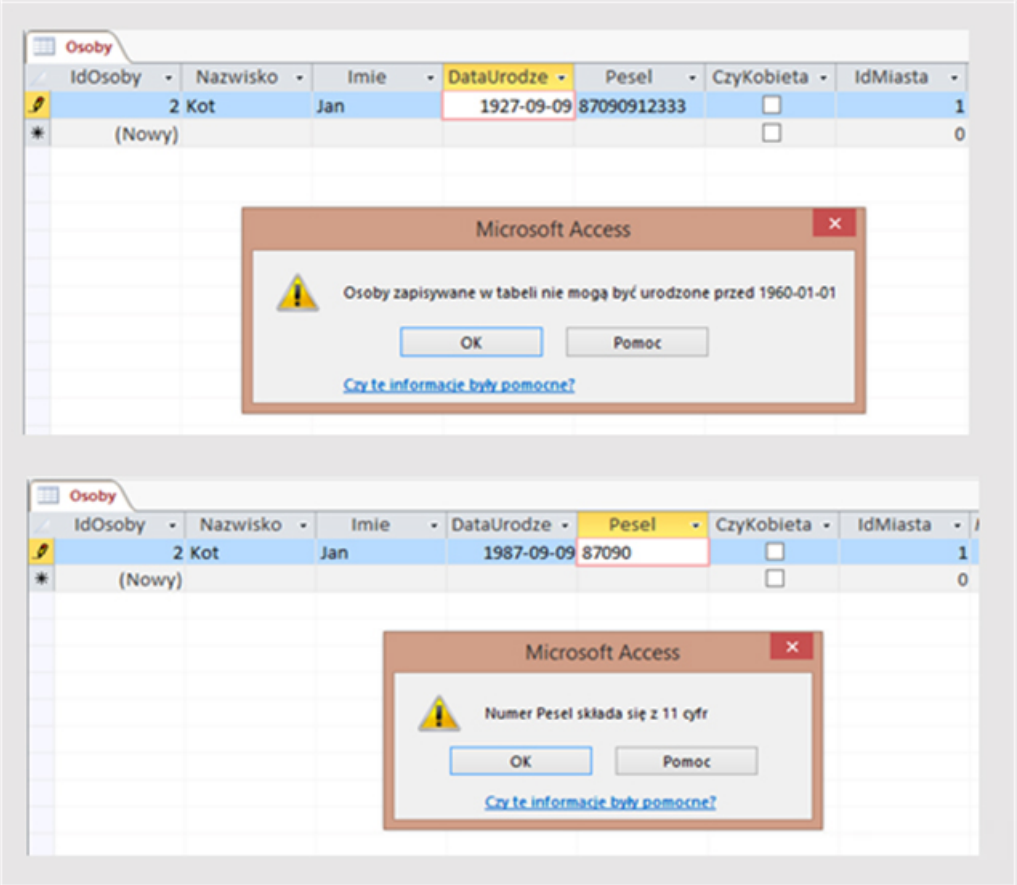
Na rysunku 6 pokazano przykładowe definicje ograniczeń dla pól tabeli `Osoby`.



Rysunek 6. Przykładowe definicje ograniczeń.

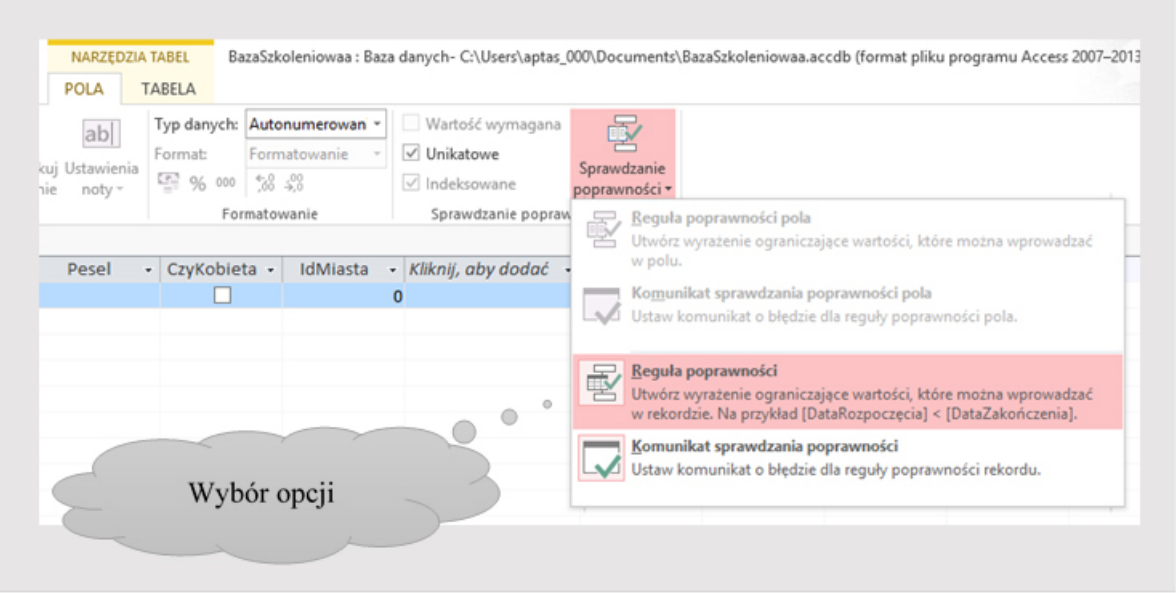
W przykładach pokazanych na rysunku 6 kolumny Pesel zdefiniowano wzorzec, określający poprawną wartość kolumny – musi składać się dokładnie z 11 cyfr. Dodatkowo, w opcji Indeksowanie, zdefiniowano brak duplikatów, czyli w tabeli nie będzie można zapisać wierszy o identycznym numerze Pesel. W programie Access można dodatkowo zdefiniować komunikat, który pojawi się przy próbie zapisania danych, które narusza określoną regułę.

Na rysunku 7 pokazano postać komunikatów przy próbie zapisania błędnych danych do kolumn Pesel i DataUrodzenia.



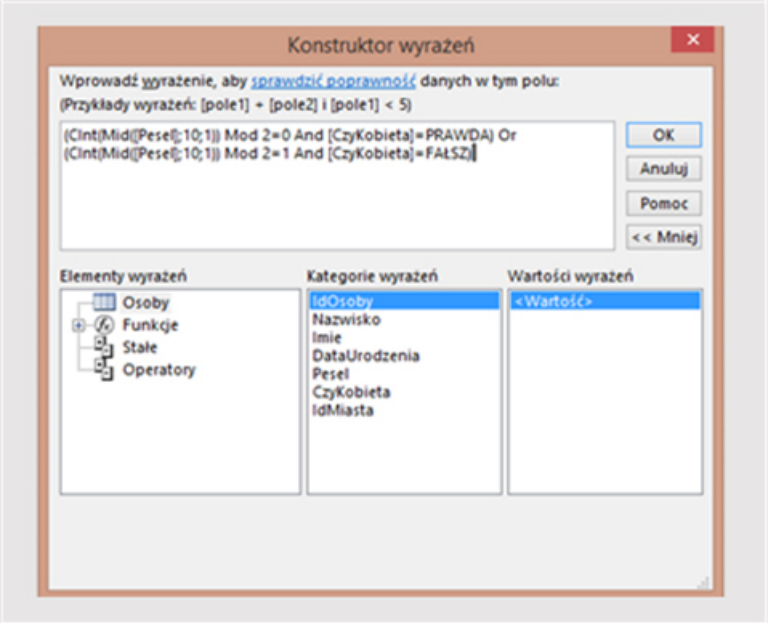
Rysunek 7. Postać komunikatów przy próbie zapisania niepoprawnych danych.

W programie Access można definiować reguły poprawności dla wiersza. W tym celu należy otworzyć tabelę – w zakładce Pola znajduje się opcja Sprawdzanie poprawności, po wyborze której wybieramy następnie opcję Reguła poprawności. Wybór opcji Reguła poprawności prezentuje rysunek 8.



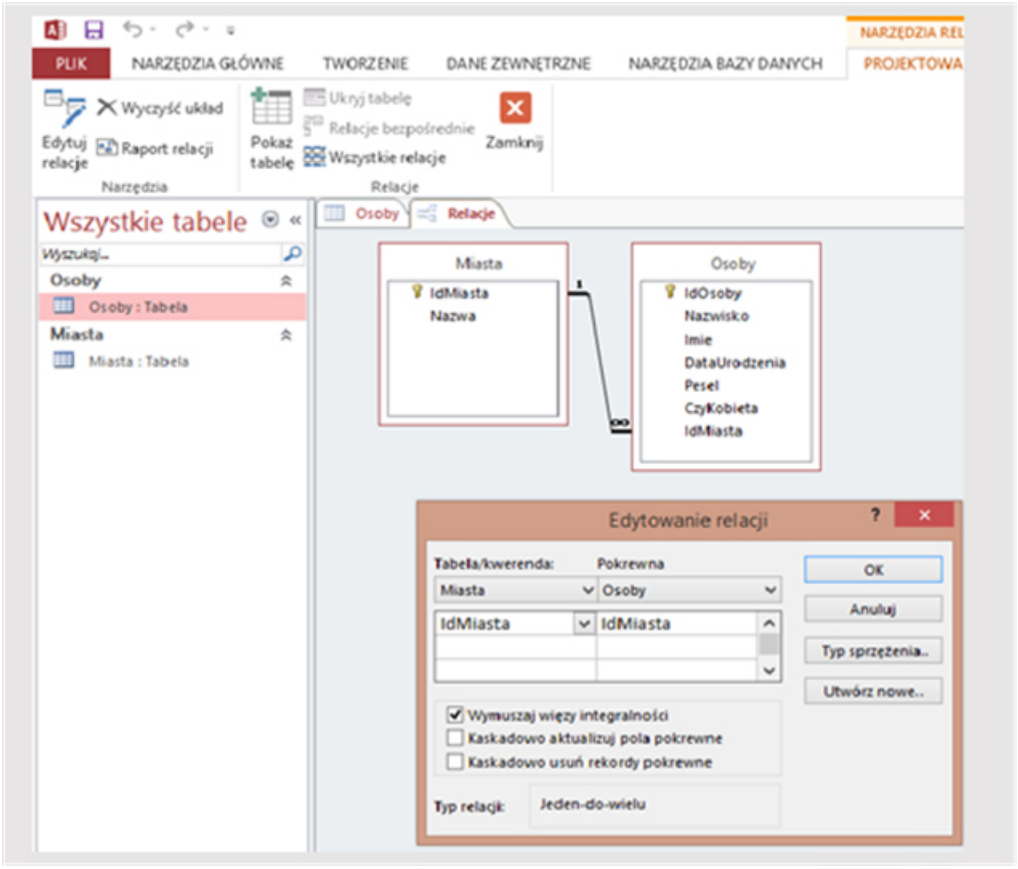
Rysunek 8. Wybór opcji Reguła poprawności.

Na rysunku 9 pokazano okno konstruktora wyrażeń, w którym zapisano regułę wymuszającą zgodność wartości w kolumnie CzyKobieta, określającym płeć osoby, z przedostatnią cyfrą numeru Pesel. Dodatkowo w opcji „Komunikat sprawdzania poprawności" można określić tekst komunikatu, który będzie pojawiał się przy próbie zapisania danych niezgodnych z regułą.



Rysunek 9. Konstruktor wyrażeń.

Kolejną możliwością zdefiniowania ograniczeń jest wymuszenie więzów integralności. Dla przykładowych tabel `Osoby` i `Miasta` chodzi o zapewnienie, by w kolumnie IdMiasta tabeli `Osoby` zapisane wartości zawsze znajdowały odpowiedni wiersz w tabeli `Miasta`. W tym celu należy utworzyć relację pomiędzy tabelami `Osoby` i `Miasta`. Na rysunku 10 pokazano definiowanie relacji.



Rysunek 10. Definiowanie relacji z wymuszeniem więzów integralności.

Po zdefiniowaniu tabel i ograniczeń wymuszających poprawność zapisywanych danych. Można przystąpić do zapisywania danych w utworzonych tabelach. W trakcie zapisywania danych będą pojawiały się komunikaty w przypadku wprowadzenia danych niezgodnych ze zdefiniowanymi ograniczeniami.

Człowiek - najlepsza inwestycja

KAPITAŁ LUDZKI
NARODOWA STRATEGIA SPÓJNOŚCI

Warszawska
Wyższa Szkoła
Informatyki

UNIA EUROPEJSKA
EUROPEJSKI
FUNDUSZ SPOŁECZNY

Projekt współfinansowany przez Unię Europejską w ramach Europejskiego Funduszu Społecznego